Documentation

Passage CMS v3.30 Configuration Manual

Date	August 2008
Status	1.0
Authors	Jasper Goertz
	Robert Lukkenaer
	Rainer Keizer

Lotus Domino is a registered trademark of IBM

Passage is a trademarks of Your Passage B.V., The Netherlands

Your Passage B.V.
Karperstraat 48
1432 PA Aalsmeer
T +31 (0)297 364001

Kamer van Koophandel 34293054
Rabobank 1366.52.670
info@yourpassage.nl
www.yourpassage.nl

CONTENTS

0 VERSIONING 5

1 INTRODUCTION (PASSAGE CONFIGURATION) 6

1.1 AIM 6
1.2 RELEVANT LITERATURE 6
1.3 THE STRUCTURE OF THIS MANUAL 6
1.4 ADDITIONAL INFORMATION 7

2 PASSAGE CONFIGURATION OVERVIEW 8

2.1 PASSAGE USER-INTERFACE AND PRESENTATION DEFINITION 8
2.2 PASSAGE CONFIGURATION DOCUMENTS 9
2.3 ACCESS TO CONFIGURATION MENU (ADVANCED CONFIGURATION) 11

3 GENERIC ELEMENTS 13

3.1 ADMINISTRATION DOCUMENT 13
3.1.1 Basic setting managed on the administration document 14
3.1.2 Keywords managed on the administration document 17
3.1.3 Library Category definition 18
3.1.4 Menu Configuration 19
3.2 IMAGE STORE 20
3.2.1 What's new view Icons (special Image store document) 20
3.3 KEYWORD LIBRARY 21
3.4 DOCUMENT TEMPLATES 21
3.5 MASTER/SLAVE CONFIGURATION 23

4 DEFINING THE SCREEN LAYOUT 24

4.1 FRAMED SITES ONLY 24
4.1.1 Standard Frame Architecture 24
4.1.2 Passage Frame definition 24
4.1.3 Additional Frame Definitions 27
4.2 PASSAGE ACTIONS 27
4.3 LAUNCHING PASSAGE FROM THE WEB-BROWSER 28

5 NAVIGATORS 30

5.1 BUTTON DEFINITION DOCUMENTS 30
5.2 CONTROLLING THE LAYOUT OF THE BUTTON THROUGH CSS 32
5.3 BUTTON IMPORT / EXPORT FACILITY 33

6 LAYOUT DEFINITION 34

6.1 GENERIC FIELDS ON THE LAYOUT DEFINITION 34
6.2 HTML LAYOUT OF THE PAGE 37
6.3 DETERMINING WHICH LAYOUT DEFINITION IS USED BY A WEB PAGE 40

6.4 CHANGING THE LAYOUT BAY PASSING A URL PARAMETER 40
7 CONTEXT DEFINITIONS 41
7.1 GENERAL INFORMATION 41
7.1.1 How it works 41
7.1.2 Determining which Context Definition is used by a page 41
7.1.3 Mandatory fields and @Formula fields 42
7.2 BASIC CONFIGURATION OF A CONTEXT DEFINITION 43
7.2.1 Fields present within each Context Object 43
7.2.2 Table block – Basics 43
7.2.3 Table block - CSS 43
7.3 CONTEXT OBJECT "NAVIGATION" 1 - 5 44
7.3.1 Table block - Placeholder 45
7.3.2 Table block - Advanced 46
7.3.3 Table block - Selection 47
7.4 CONTEXT OBJECT "NAVIGATION TRAIL" 48
7.4.1 Table block - Placeholder 48
7.4.2 Table block - Advanced 48
7.5 CONTEXT OBJECT "VIEW" 1 - 10 49
7.5.1 Table block - Placeholder 49
7.5.2 Table block - Advanced 49
7.5.3 Table block - Selection 49
7.5.4 Table block - Display 50
7.5.5 Available URL parameters 52
7.6 CONTEXT OBJECT "RELEVANT DOCUMENTS" 1 - 3 53
7.6.1 Table block - Placeholder 53
7.6.2 Table block - Advanced 54
7.6.3 Table block - Selection 54
7.6.4 Table block - Display 55
7.6.5 Available URL parameters 56
7.7 CONTEXT OBJECT "COMPUTED TEXT" 1 - 5 56
7.7.1 Table block - Placeholder 57
7.7.2 Table block - Advanced 57
7.7.3 Table block - Computed Text 57
7.8 CONTEXT OBJECT "IFRAME" 1 - 5 58
7.8.1 Table block - Placeholder 58
7.8.2 Table block - Advanced 58
7.8.3 Table block - IFrame 58
7.9 CONTEXT OBJECT "IMAGE" 1 - 10 60
7.9.1 Table block - Placeholder 60
7.9.2 Table block - Advanced 60
7.9.3 Table block - Image 61
7.10 CONTEXT OBJECT "SITEMAP" 62
7.10.1 Table block - Placeholder 62

7.10.2 Table block - Advanced 62
7.10.3 Table block - Selection 63
7.11 CONTEXT OBJECT "SEARCH RESULTS" 64
7.11.1 Table block - Placeholder 64
7.11.2 Table block - Advanced 64
7.11.3 Table block - Display 65
7.11.4 Available URL parameters 67
7.12 TABLE BLOCK - OVERVIEW OF USED OBJECTS 68
7.13 PASSING PARAMETERS TO CONTEXT OBJECTS 69
7.14 OVERRIDING THE DEFAULT BEHAVIOUR OF CONTEXT OBJECTS 70

8 WEB FRONT END FORMATTING 71

8.1 JAVASCRIPT LIBRARY 71
8.2 CSS DOCUMENTS 72
8.2.1 Centering a page 72
8.3 JAVASCRIPT VIEWS 73

9 MAIL-IN FILTERS 75

9.1 INTRODUCTION 75
9.2 DEFINING MAIL-IN FILTERS 76
9.3 STRUCTURED MAIL IMPORT 76

10 DOCUMENT LIFE CYCLE MANAGEMENT 78

11 CUSTOMIZING PASSAGE CODE 79

12 CONFIGURING THE DATABASE / SECURITY 80

0 Versioning

Table 1: Versioning

Version	Date	Author	Change
1.0	August		Initial version Passage CMS 3.30

1 Introduction (Passage Configuration)

1.1 Aim

This manual aims to provide a solid knowledge base for the configuration and management of a Passage site. It has been written for all those involved in these activities, like graphic designers, system managers, information brokers and system integrators.

1.2 Relevant Literature

This manual is not intended as an instruction manual for an intra-, extra- or internet end-users. A customised employee education program will need to be devised for each particular organisation. The manual is not intended for content managers either. Content managers are referred to the end-user manual. The structure of this manual is based on the sequence of operations involved in configuring Passage. We are assuming that Lotus Notes, the Domino server and the Passage Database(s) have been installed. Readers are expected to have basic knowledge about working with the Internet and some basic HTML knowledge.

1.3 The Structure of this Manual

	This manual has the following structure:
Chapter 1	General introduction about Passage
Chapter 2	Passage configuration overview
Chapter 3	The site's administrative document. This document is used to predefine a number of the basic features of an Internet or Intranet site, like whether it will be possible to order brochures via the site and all the keywords to be used in the pull-down lists.
Chapter 4	Defining the screen layout
Chapter 5	Navigators/Buttons. How do users navigate Passage?
Chapter 6	Layout: How the content of a Passage database is presented on the Web.
Chapter 7	Web front end formatting issues
Chapter 8	Mail-In filters
Chapter 9	Document life cycle management
Chapter 10	Custom fields in Passage
Chapter 11	Database ACL settings

1.4 Additional Information

An alternative source of information is the Passage support website: http://www.yourpassage.nl/passage and Passage end-user and administration manuals.

2 Passage configuration overview

The Passage configuration determines the functionality of the database. Configurable items of the Passage database are:

Item	**Description**
Information types	Definition of content streams, keywords, functions, mainly done on the Administration document
User interface, Navigation	Definition of the navigational structure of the site, mainly done in framedefinitions and navigators (button documents)
Presentation (Graphic design)	The presentation of the database to the end users is defined by: • Layout definitions • Context Definitions • JavaScript • CSS pages • Image store documents
Workflow	Workflow settings are done on the databases access control list (ACL), as well as groups in the NAB
Security settings	Access rights, read/write, create/delete settings are done on the databases access control list (ACL). The delete rights for web-editors is set on the Administration document.

2.1 Passage user-interface and presentation definition

Passage CMS has many options on how to present the data to the End user. The basic split can be made between "Frame" and "Frameless" sites.

For "Framed" sites:
The Passage configuration has the following data model:

- Each database has one or more framesets
- Each frameset has up to 4 frame content definitions (red, green, blue and gold)
- Each frame content definition relates to 1 content type (document, view, button navigator, URL)
- Each content type has it's own layout definition

- Each layout definition links it's own JavaScript and CSS definition documents

For "Frameless" sites:
The Passage configuration has the following data model:

- The screen layout is defined by splitting the Screen in one or more <DIV> tags. The <DIV> tags are positioned using CSS.
- Each content type is connected to a layout definition type (document, view, button navigator, URL)
- Each layout definition contains the <DIV> set, and links to the corresponding CSS Definitions and JavaScript files. Frequently, additional JavaScript Pages are added to create menu effects based on Button & Library documents.

2.2 Passage configuration documents

Necessary documents in a clean (new) Passage database:

ACL	Correct access settings
Administration or General document	1 always necessary, containing database wide category and keyword definitions
Frame definitions	1 definition with the name MAIN must be present (with Frame Sites, in Frameless Sites no frame definitions needs to be present)
*Layout definitions***	The following layout definitions must be present: - Default for documents - Depending on the Passage elements used, the following layout definitions must be present: - Default for navigators - - Default for views – Library Library View Mail Search results $$Search When using Passage for a frameless website the use of a "- Default for documents –" and a "Library" layout definition is sufficient.

*Context Definitions**	If no context definition is available the functionality is ignored and won't be used.
*JavaScript**	JavaScripts are used for effects on the screen, menu's, view formatting, error checking etc. there is a large number of JavaScript functions that can be used in Passage sites. Limitations of Javascripts are the poor debugging possibilities, and the fact that all processing is done in the client side. // JavaScripts delivered in the Passage databases or Templates are delivered AS IS. You are obliged to respect and act in accordance to the Copyright statements of previous users.
*Text Pages**	Depending on the Passage elements used, the following text pages need to be present: · 00 (return page after mail submission) Thank you, your message has been received · 01 (return page after a Response is submitted) Thank you, your response has been received · 02 (return page after a Request is submitted) Thank you, your request has been received · 03 (return page after a Presentation is uploaded) Thank you, your presentation has been received · Access Denied (when accessing a page with a non-published status) - This information can not be accessed from the web! · Not Available (page is not available) Not available
Master / Slave configuration	In Large Passage Implementations usually the configuration is stored in a central repository database (the Master) and the Content Databases (the Slaves) depend on the master. This decreases the amount of configuration time enormously.

* Only necessary if certain functionality is used

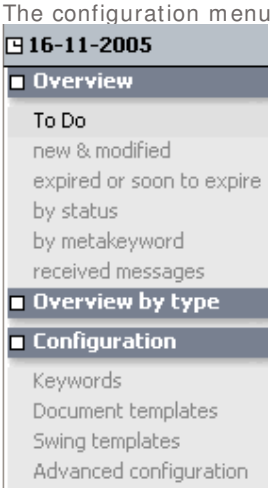
** It is possible to create one layout definition that is eligible for all types mentioned

2.3 Access to Configuration Menu (Advanced configuration)

Configuration menu

To change the structure or the appearance of the Intranet or website you need access to the advanced configuration menu.
This access is given to you by the system administrator by setting the right role (**ACL Role: ConfigEd and Access level must be Editor**).

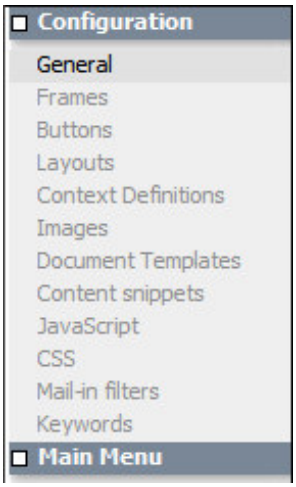
The configuration menu is shown in the bottom left of the menu.



Users obviously require certain rights to access the configuration navigator (**ACL Role: ConfigEd and Access level must be Editor**)

"Master" configuration menu

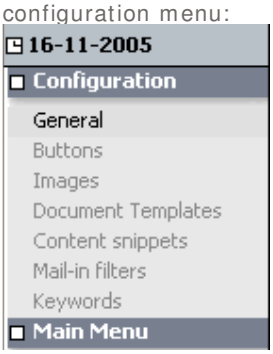
All configuration possibilities are displayed when the master menu is active:



Warning: changing the configuration changes your site. Beware. Create backups before doing this.

"Slave" configuration menu

A subset of the configuration functions is displayed with the Slave configuration menu:



3 Generic elements

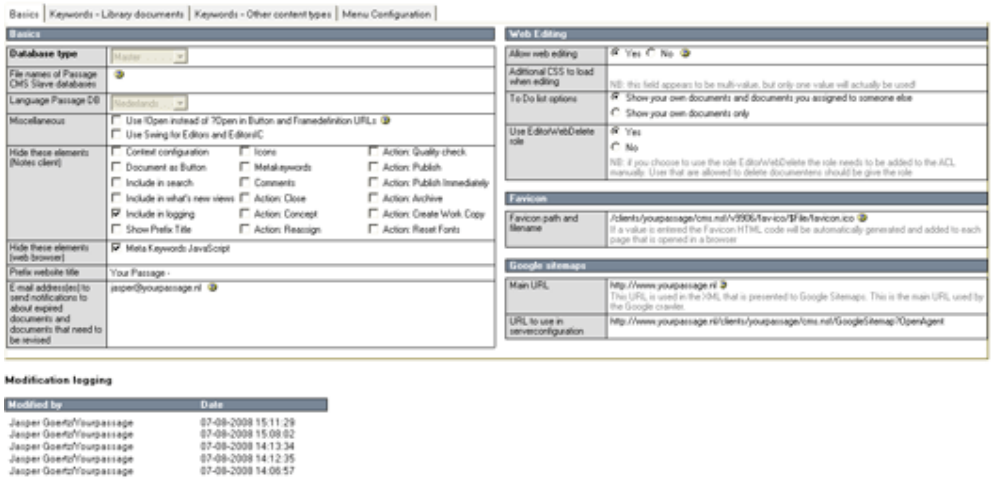
The configuration of the Passage content management system can be divided into 2 parts, generic elements and specific elements.
This chapter describes the generic elements, being:

Administration document	The data-model of the Passage Content Management system, specifying information types, keywords, general settings, etc .
Image store	Documents containing generic images, to be used anywhere in the site.
Keyword library	Documents containing keywords, to be used by custom developments, library documents, meta-information, etc.
Templates	Pre-defined documents that can be pasted into rich text fields, while inheriting styles and layout definitions.
Snippets	Pre-defined pieces of content that can be pasted into rich text fields.

3.1 Administration document

The Passage content management system has a very rudimentary data dictionary. The content types in Passage are pre-defined (except for the custom subforms that can be made by very advanced users/developers with a Lotus Notes developer client).

Selecting the Administration document from the Passage advanced configuration menu displays the following Image:



The data-dictionary is managed via the Administration document, this document contains:

- keywords per data-type

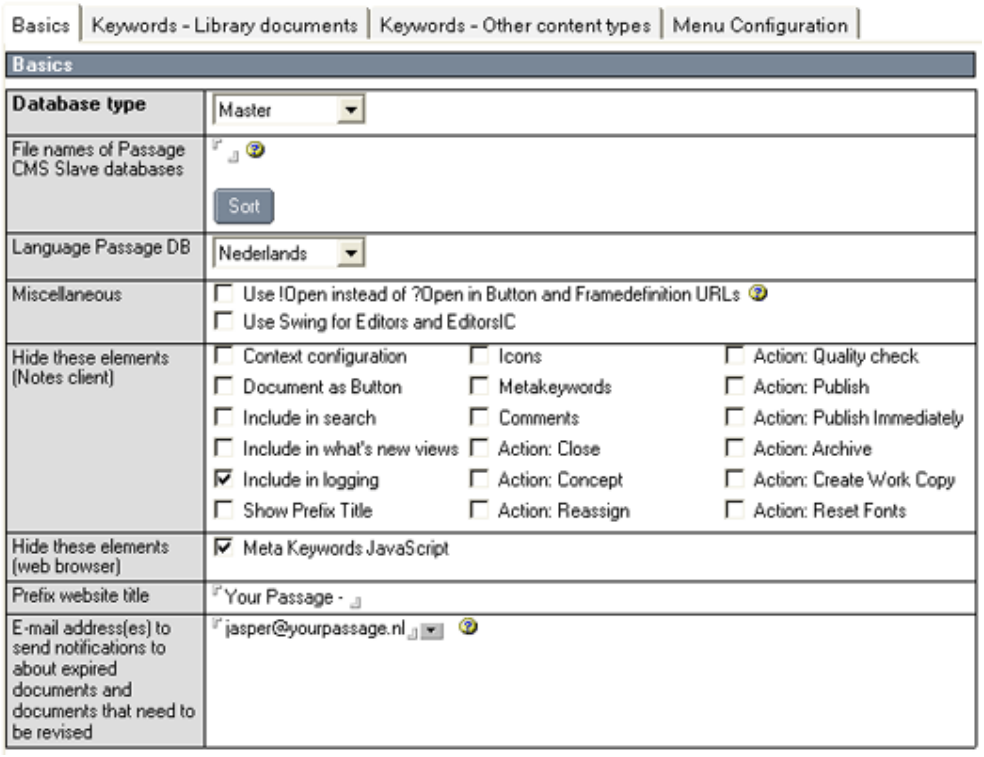
- when applicable, mail-addresses for users handling specific events (mail-forms)
- database language, enable web-editing
- Master / Slave setting
- Hide workflow elements
- Web Editing settings
- Other general settings: e.g. Google sitemaps, Favicon, etc.

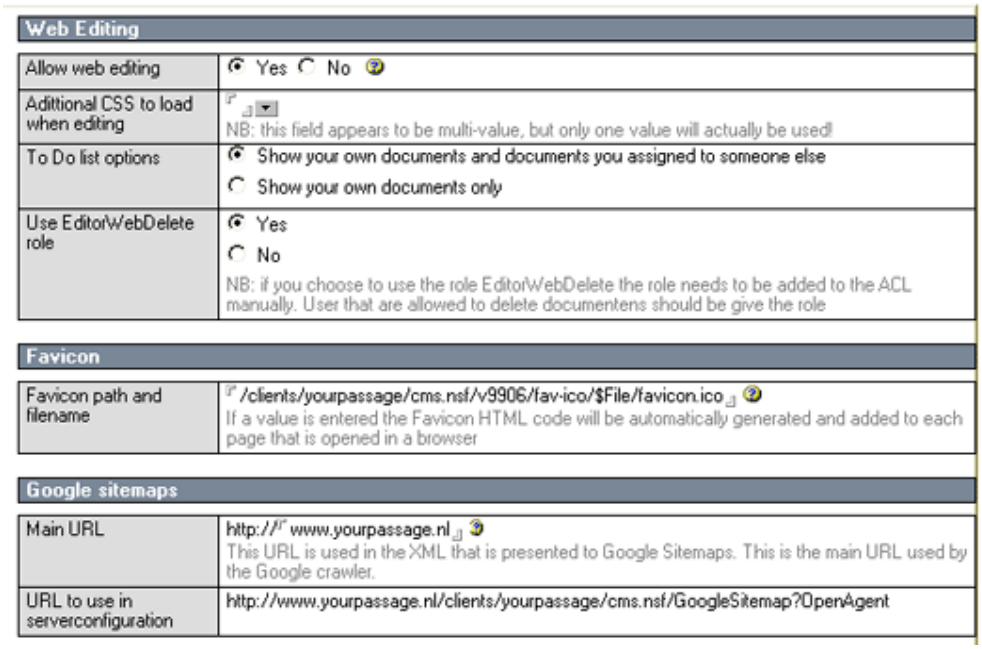
A special data-type in Passage is the library document type. The configuration can define a number of information categories (this used to be 31, but this limitation is removed, do not use more than 50 library categories for realistic usage).
For each information category the configuration can specify a set of keywords in the general document (keyword 1 & 2). This function is present for backward compatibility, it is better to use separate Keyword documents.

All functions on the Administration document are "self explanatory" and when applicable accompanied by a Helper text.

3.1.1 Basic setting managed on the administration document

The basics tab of the administration contains a number of settings the influence the appearance and available functions all through the Passage database.




Web Editing
Allow web editing
Yes
No
Additional CSS to load when editing
NB: this field appears to be multi-value, but only one value will actually be used!
To Do list options
Show your own documents and documents you assigned to someone else
Show your own documents only
Use EditorWebDelete role
Yes
No
NB: if you choose to use the role EditorWebDelete the role needs to be added to the ACL manually. User that are allowed to delete documentens should be give the role
Favicon
Favicon path and filename
/clients/yourpassage/cms.nsf/v9906/fav-ico/$File/favicon.ico
If a value is entered the Favicon HTML code will be automatically generated and added to each page that is opened in a browser
Google sitemaps
Main URL
http://www.yourpassage.nl
This URL is used in the XML that is presented to Google Sitemaps. This is the main URL used by the Google crawler.
URL to use in serverconfiguration
http://www.yourpassage.nl/clients/yourpassage/cms.nsf/GoogleSitemap?OpenAgent

The following settings are managed on the tab:

Basics	
Database type	Choose “Master” for a standalone application or when the design of this database must be reused by other databases (slaves). Choose “Slave” when this database should derive it’s layout, and other functions, from a different (master) database.
File names of Passage CMS Slave databases	If there are database that reuse components from this database (slaves) there file name and path need to be placed here.
Language Passage DB	Choose the language for the labels and menus in/for the edit environment. This also determines the language of the labels of the default digital forms on the website: contact, forward to a friend, vacancy.
Miscellaneous	Optional settings
Hide these elements (Notes client)	Optional settings. Determines which functions are available in the Notes client.
Hide these elements (Web browser)	Optional settings. Determines which functions are available in the web browser.

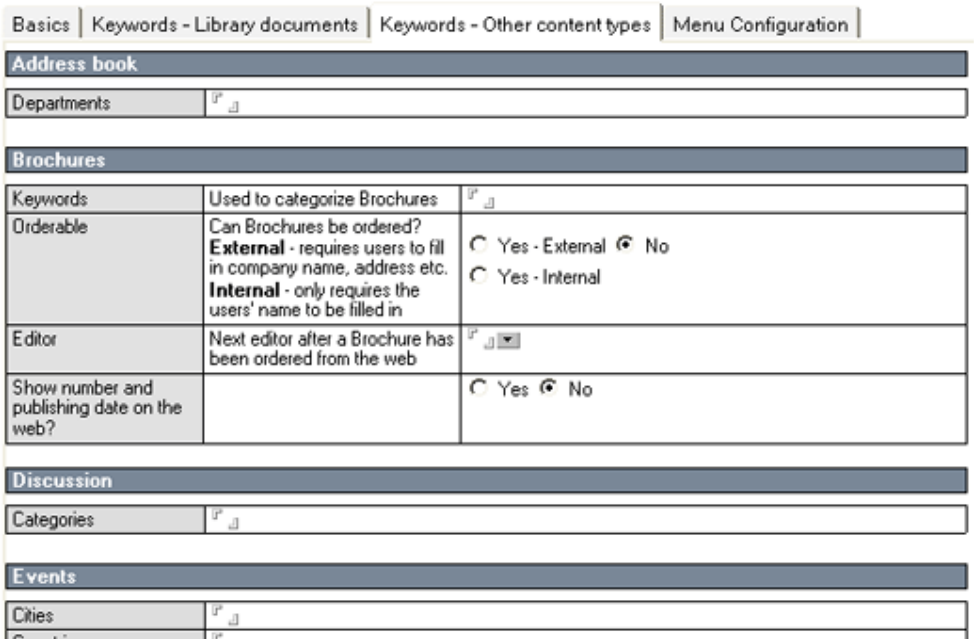
Prefix website title	Optional setting. Can be used to provide a prefix to the website title shown in the browser title bar.
E-mail address(es) to send noti...	When documents expire or need to be revised, an e-mail notification will be sent to the filled in e-mail addresses.
Web Editing	
Allow web editing	Choose "Yes" to enable web editing (using the web editing environment via the web browser).
Additional CSS to load when editing	Optional setting. When needed, an extra CSS style sheet can be defined that is used when a document is in edit mode in the web browser. The CSS style sheet chosen here extends/overwrites all other (on the layout definition and/or context definition used) chosen CSS style sheets.
To Do list options	Choose which documents to show in the To Do list of the webedit environment.
Use EditorWebDelete role	Everyone with the role EditorIC is allowed to delete all content. Some implementations demand that EditorIC's need to be able to publish and archive all content in the database, but won't allow the deletion of all documents. The ACL "delete" property in the ACL is ignored. Therefore an extra role was added. If you use the EditorWebDelete documents can only be deleted by persons who have this role.
Favicon	
Favicon path and filename	Optional setting. Adding the full path to a favicon file (*.ico) will show a favourites icon in the address bar of the browser, etc. If empty, the field is ignored.
Google sitemaps	
Mail URL	The Passage cms automatically generates an XML sitemap. In order for the to work correctly, the system needs a full an absolute path. It is possible that your websites can be reached using multiple URL's, choose the most important URL.
URL to use in server-configuration	If the main URL is filled in, the system will show an URL that can be copied to the server configuration.

3.1.2 Keywords managed on the administration document

Each information type (specified below), except for the library documents, has specific fields that determine the functionality. These fields are:

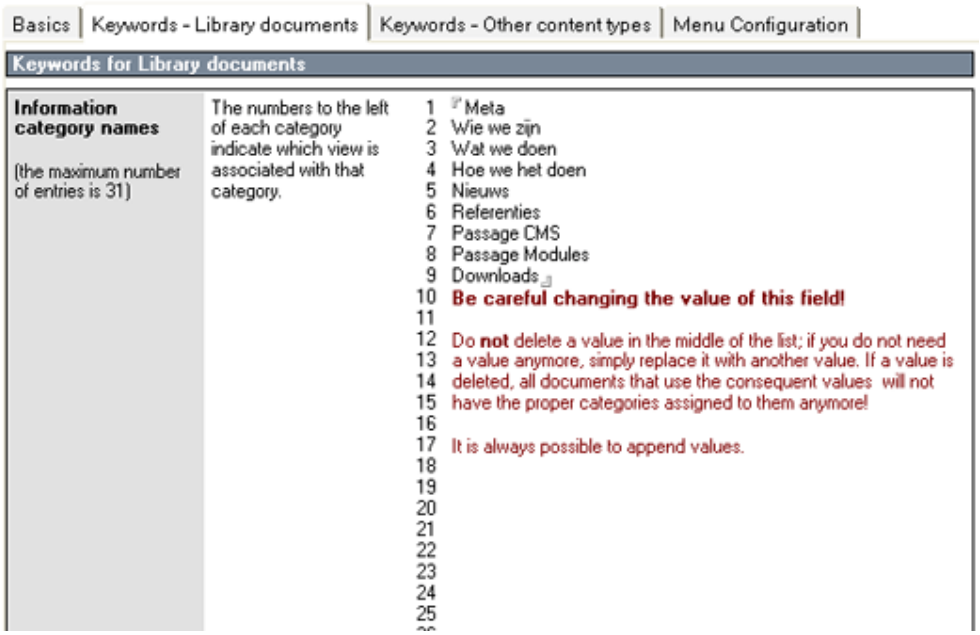
- keywords
- mail-addresses
- some keyword dependent types

These keywords are managed on the administration document.



3.1.3 Library Category definition

On the Administration document the library categories can be defined. In versions prior to 3.00 this was limited to 31, current versions do not have this user limit.


Basics
Keywords - Library documents
Keywords - Other content types
Menu Configuration
Keywords for Library documents
Information category names
(the maximum number of entries is 31)
The numbers to the left of each category indicate which view is associated with that category.
1 Meta
2 Wie we zijn
3 Wat we doen
4 Hoe we het doen
5 Nieuws
6 Referenties
7 Passage CMS
8 Passage Modules
9 Downloads
10 Be careful changing the value of this field!
12 Do not delete a value in the middle of the list; if you do not need
13 a value anymore, simply replace it with another value. If a value is
14 deleted, all documents that use the consequent values will not
15 have the proper categories assigned to them anymore!
17 It is always possible to append values.

The number of specific views (by name / by keyword) remains limited to 31.

!!! Do NOT renumber !!!

Library categories are defined by number and by name. When defining library categories, it is very IMPORTANT, that you do not change or renumber the library categories, since the views use a name/number conversion to select documents (done for performance reasons). Renumbering the library definitions without taking precautions will definitely result into jibberish. Instead of making the values blank, replace them with dummy values. All values need to be unique!

If you do not use the predefined Library views the above can be ignored. Changing the values is no problem.

3.1.4 Menu Configuration

Passage offers the possibility to configure the navigation of the Passage database in the Notes client. On the tab “Menu Configuration” you’ll find the code that is used to create the menu in the edit environment.

The “Reset menu” will create buttons for all the default content types as well as the specific configured library types. Clicking “Update menu” will replace the existing buttons with the buttons defined by clicking the “Reset menu” button. It is possible to remove certain buttons from the code before clicking the “Update menu” button, hereby removing them from the menu.

In Passage CMS 3.0, a minimal ACL access level of “Designer” was needed to successfully execute the action. From version 3.10 and up the actual updating is done by an agent in the backend, the minimal ACL access level of designer is no longer necessary.

3.2 Image store

The image store is used to archive all graphic elements. The images are stored in several image store documents identified by name.

For example 'drempels weg' functionality images are stored in this image store:

3.2.1 What's new view Icons (special Image store document)

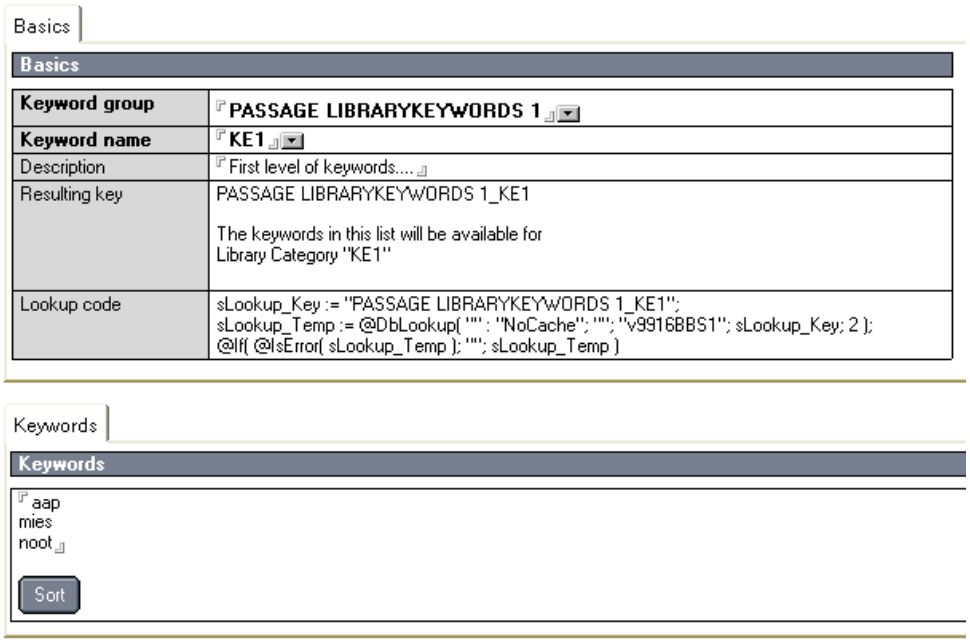
Each document has a field called "Icons". The field makes it possible to select images from a special image store document with the following key: PASSAGE_ICONS. This document is a regular image store document but has a reserved name.

The images chosen in the "Icons" field can easily be used is views, on documents, etc. there is a special what's new view that allows adding images or icons to the documents shown in the what's new view.

3.3 Keyword library

The keyword library is used for storing extra sets of keywords. This can be useful for customising Passage.

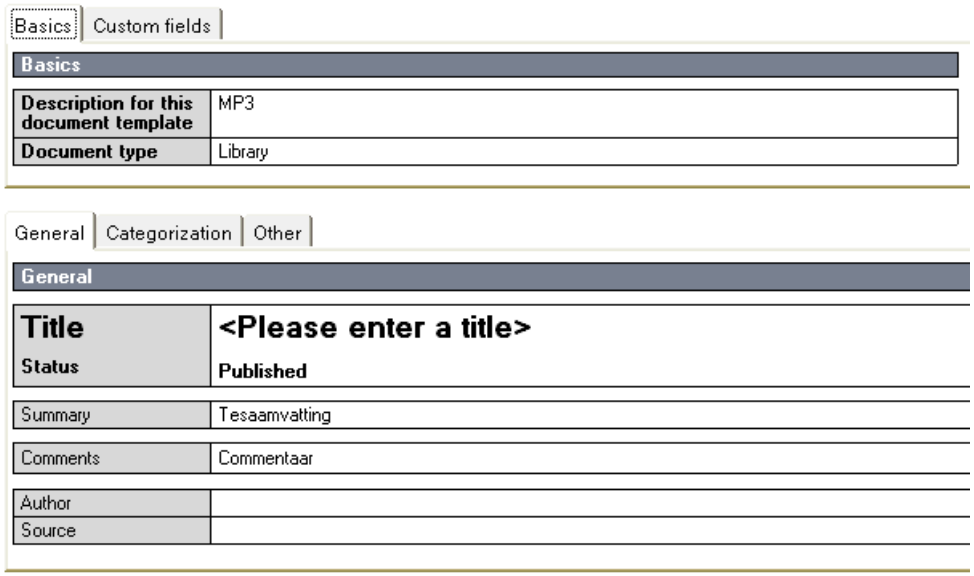
Meta keywords for the sitemap functionality and relevant documents are also stored here. Meta keywords may not contain any spaces. Replace a space by an underscore.



3.4 Document Templates

Using a document template is the only way documents can be created in Passage CMS (using the Lotus Notes client). Templates are pre-defined documents that guide the editors to correct output that guarantees performance and consistency.

You can create your own template using the template functionality. Templates are synchronized with the Passage configuration at creation time.



Basics | Custom fields

Basics

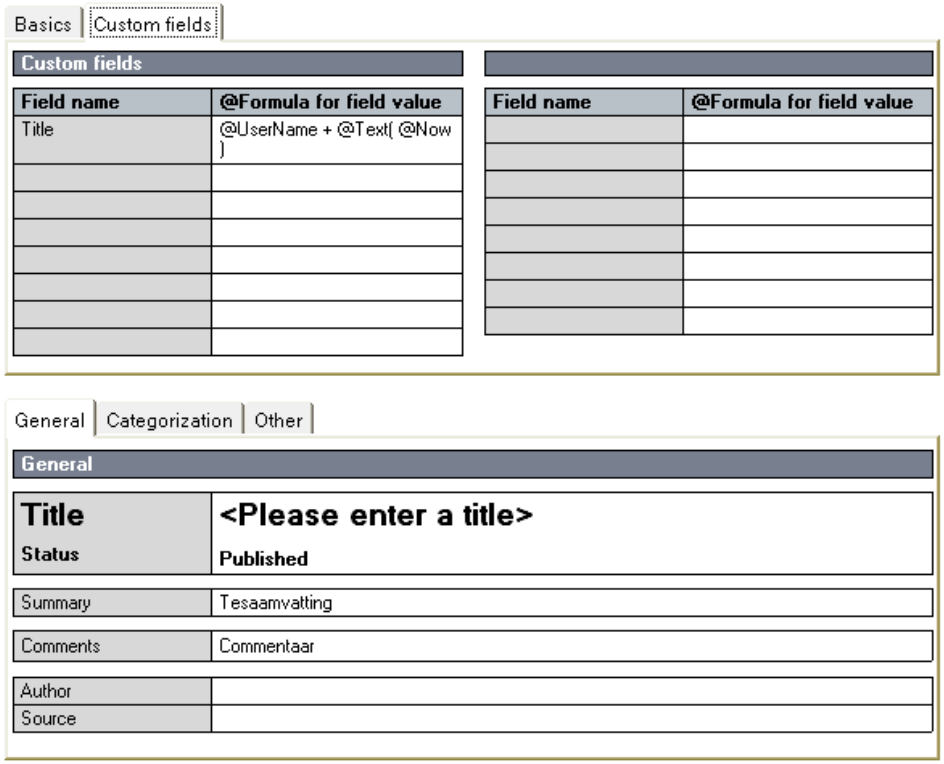
Description for this document template	MP3
Document type	Library

General | Categorization | Other

General

Title **Status**	**<Please enter a title>** **Published**
Summary	Tesaamvatting
Comments	Commentaar
Author	
Source	

Using the custom field's functionality, you can create your own, advanced templates, using evaluated @formulas.



Basics | Custom fields

Custom fields

Field name	@Formula for field value
Title	@UserName + @Text(@Now)

Field name	@Formula for field value

General | Categorization | Other

General

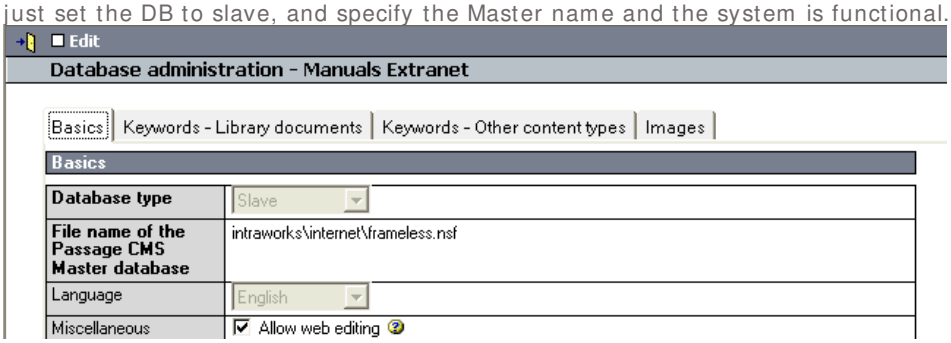
Title **Status**	**<Please enter a title>** **Published**
Summary	Tesaamvatting
Comments	Commentaar
Author	
Source	

3.5 Master/Slave configuration

In the Master/Slave configuration (from Passage 1.83 and higher), multiple Passage databases are specially configured such that they share configuration and design elements. The following table shows the difference between Master and Slave.

Master	Slave
Administration document	Administration document
Buttons	Buttons
Images	Images
Templates	Templates
Mail-In filters	Mail-In filters
Keywords	Keywords
Framesets	
Layouts	
JavaScripts	
CSS	

Master Slave configuration is controlled on the Database Administration Document, just set the DB to slave, and specify the Master name and the system is functional.



Pitfalls:
When using JS views the routines need to be changed to specify the Slave database (this is done by changing the variable DBNameHTML into DBNameHTML_C).
Only one type of frameset out of all framesets can be used.

Advantages are:
Easy configuration and modification
Speed since fewer documents are used (caching is optimised).

Available from: Passage 1.83 ML

Top of your administration document where you have to select if you want a master or a slave database.

4 Defining the screen layout

4.1 Framed sites only

To use frames the Passage frame definition must be configured. If you choose to create a frameless site, configuration of the Passage frame definition is not necessary.

4.1.1 Standard Frame Architecture

Passage uses a maximum of 4 frames to structure the website. The Passage frame architecture organises navigation elements hierarchically. There are three different navigation elements:

1. Top navigators (comparable with chapters)
2. Cluster navigators (comparable with paragraphs)
3. Extra or generic Buttons (always available, for example: 'search', 'what's new' and 'mail').

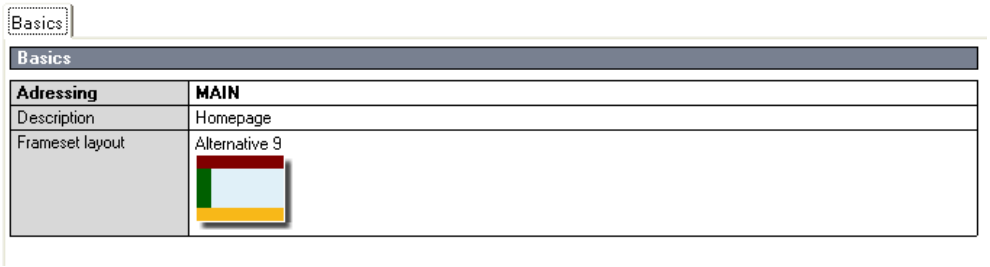
The fourth frame in Passage is being used to publish the content.

New from version 2.0 is the possibility to inject HTML Code Pre- and After the Frameset, thus giving the opportunity to create framesets around the frameset (e.g. for centered sites).

4.1.2 Passage Frame definition

Any number of framedefinintions can be defined in the Passage database. Normally one or 2 framedefinitions are sufficient.

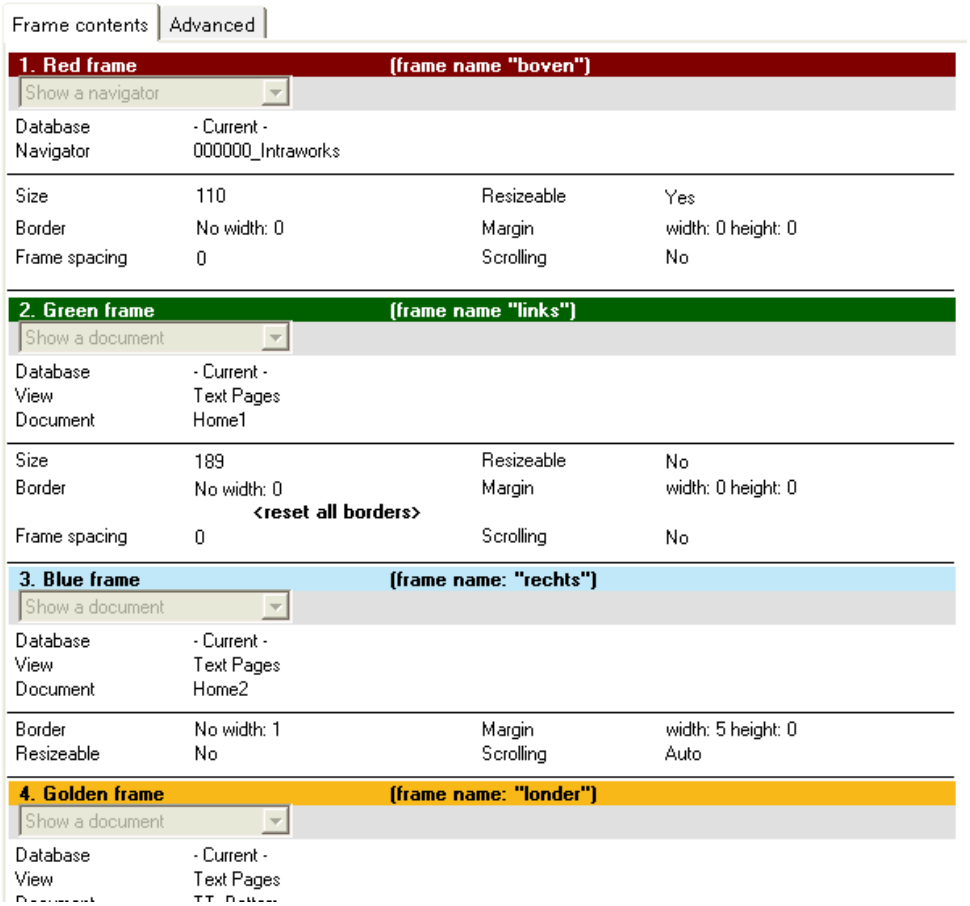
In a Framed Passage site there must be one frame definition with the name: **MAIN.** This framedefinition is used in the buttons.



Frame title and Meta Keywords:

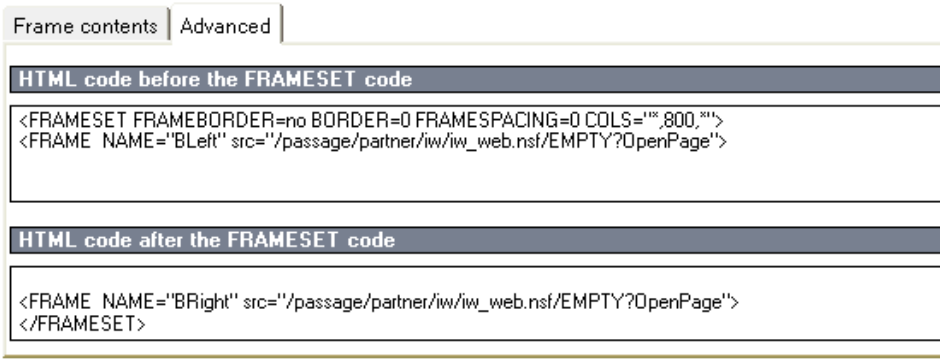
Site title and META tags	
HTML Title	Intraworks
META description	Welkom bij Intraworks!
META keywords	Content Management, Intranet, Internet, Lotus Notes, Lotus Domino, Lotus, Notes, Domino, groupware, Passage, Projects, ASP, Consultancy, Consultants, Kennisoverdracht, Nieuwe Media, Nieuwe, Media Communicatie, Samenwerken, Amsterdam, Nederland, Training, Coaching, Advies, CLP, intranet, lotus domino, lotus notes, notes 5.0, notes 4.6, notes r5, business suite, development environment, itf software development kit, itfsdk, itf sdk, itf business partners, lotus business partners, itf.community, itfcommunity, itf community members, itf development tools, buy software tools, buy software solutions, colaboration, colaborative, collaborative ebusiness solutions, collaborative e-business solutions, collaborative ebusiness applications, collaborative e-business applications, collaborative business solutions, business to business collaboration, business-to-business collaboration, b2b, itf business suite, itf architected applications, itf-architected applications, lsx, business components, developer tools, business software shop, developer component e-shop, developer component store, component-based development, component based development, rapid application development, add-ons, add-ins, reusable components, re-usable, reuseable, freeware, shareware, api, application programming interface, itf isv, itf asp, lotus isv, lotusphere, workflow, e-crm, The foundation of E-business

Defining the Frame contents can be done using the colour identification. The element top open is helped.



HTML Code injector:

HTML Code can be injected in the Frame set. This creates many possibilities, but you need to have HTML Knowledge.



Google Compatibility

The Frameset generates code that is accepted by Google (nov 2005) for indexation. This feature can change obviously.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<SCRIPT language="JavaScript1.2"
src="/passage/partner/iw/iw_web.nsf/PassageJSLibrary.js?OpenPage"
type="text/javascript"></SCRIPT>
<SCRIPT language="JavaScript" type="text/javascript">
DBNameHTML="/passage/partner/iw/iw_web.nsf";
DBNameHTML_C="/passage/partner/iw/iw_web.nsf";
</SCRIPT>
<!--
Created using Passage CMS
-->
<TITLE>Intraworks</TITLE>
<META NAME="description" content="Welkom bij Intraworks!">
<META NAME="keywords" content="Content Management, Passage Projects, Intranet, Internet,
Lotus Notes, Lotus Domino, Lotus, Notes, Domino, groupware, Passage, Projects, ASP,
Consultancy, Consultants, Kennisoverdracht, Nieuwe Media, Nieuwe, Media Communicatie,
Samenwerken, Amsterdam, Nederland, Training, Coaching, Advies, CLP">
</HEAD>
<FRAMESET FRAMEBORDER=no BORDER=0 FRAMESPACING=0 COLS="*,768,*">
<FRAME FRAMEBORDER=0 SCROLLING=No MARGINWIDTH=0 MARGINHEIGHT=0 NORESIZE
NAME="BLeft" SRC="/passage/partner/iw/iw_web.nsf/v01013/TT_Blank!open">
<FRAME SCROLLING=No MARGINWIDTH=0 MARGINHEIGHT=0 NORESIZE NAME="BContent"
SRC="/passage/partner/iw/iw_web.nsf/framedefinitions/main!OpenDocument">
<FRAME FRAMEBORDER=0 SCROLLING=No MARGINWIDTH=0 MARGINHEIGHT=0 NORESIZE
NAME="BRight" SRC="/passage/partner/iw/iw_web.nsf/v01013/TT_Blank!open">
</FRAMESET>
<A HREF="/passage/partner/iw/iw_web.nsf/v14013!openview">All documents for
crawler</A>
</HTML>

<!--</SCRIPT>
<SCRIPT LANGUAGE="JavaScript1.2"
src="/passage/partner/iw/iw_web.nsf/p0002JVS3?OpenPage"></SCRIPT>
```

4.1.3 Additional Frame Definitions

Your Passage can develop additional frame layouts on request. Please contact your representative for details.

4.2 Passage actions

Passage Actions are executed from within a frame or button document. Each action performs a specific action on a notes/passage database or onto another system. Actions are:

Show document	Shows a document from a view. Opens a document from the specified database that is located in the specified view. In the second row you now need to enter which database and view the document comes from.
Create document	Creates a specific document in a Notes database

Show a view	Shows a view from a notes database. View option: Expand (Feature that expands any addressed views) and collapse (Feature that collapses any addressed views)
Show a navigator	Shows a button navigator (within Passage)
Open a database	Opens a database with the default database launch options
URL	Anything that can be addressed from the client's system
Open the users own mail file	Opens the users own notes web mail file (only in Passage Intranet version!)

The Passage program handles the correct translation of the above actions into HTML commands.

4.3 Launching Passage from the web-browser

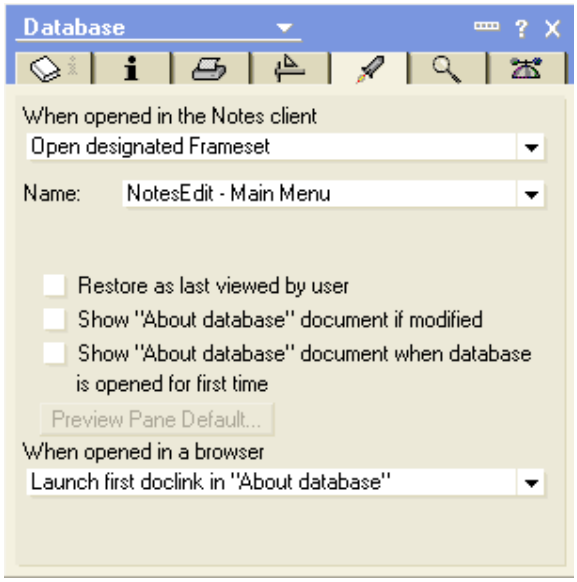
The Passage database needs to be configured to perform the right action when the database is opened from the web-browser. The only correct action for Passage used to be to open the MAIN frame definition when the database is launched.

Nowadays this is no longer necessary. Passage can open any content element and position a frameset (if required) around the content.

To perform the initial action:

This is configured as follows:

1. Open the Passage database.
2. Right click on the frameset, text page or other document.
3. Choose "Copy as doclink".
4. Open the database in the Lotus Notes designer client.
5. Open the "About this database" (can be found under "Other") document. Paste the doclink in the "About this database" document. Make sure it is the first doclink.
6. Save the "About this database document".
7. Select the database properties box, and select the launch tab (see image below).


Database
When opened in the Notes client
Open designated Frameset
Name:
NotesEdit - Main Menu
Restore as last viewed by user
Show "About database" document if modified
Show "About database" document when database is opened for first time
Preview Pane Default...
When opened in a browser
Launch first doclink in "About database"

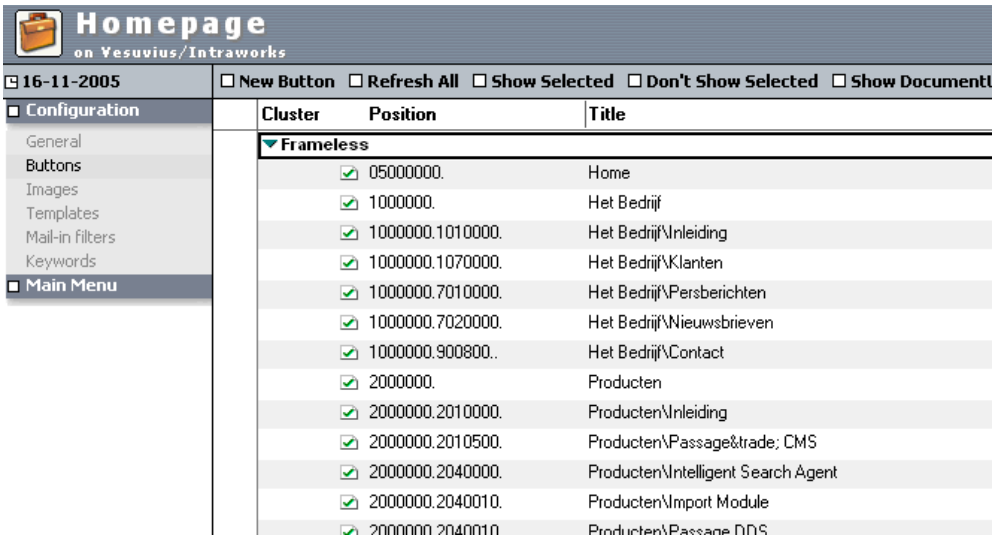
8. From the database launch option “When opened in a browser”, select launch first Doclink in "About database". The database will now launch the mainframe definition when the database is opened from the web.

PS: you’ll need Designer access to perform this operation.

5 Navigators

Passage navigators are used to navigate a Passage site. Navigators establish the hyperlinks required to access new navigators and other functionality.

Within Passage, navigators are created by grouping button documents by keyword (buttons are explained in the following paragraphs). Using these keywords, passage navigators can be addressed from Frame definitions or from other buttons documents. The following picture shows the button selection and the representation on our website.



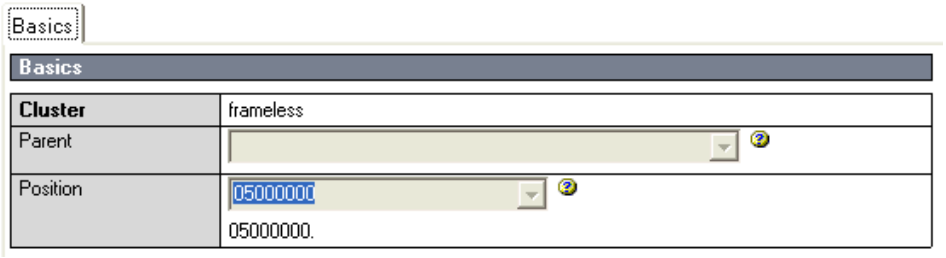
5.1 Button definition documents

Buttons are a very flexible way of creating a navigation structure in your I-net. Passage Projects is continuously developing new methods to increase the usage of buttons. Please contact Passage Projects on a regular basis for new developments. On each button you can define the following variables:

- Positioning
- Appearance
- Actions
- Audience (in Passage intranet version only)

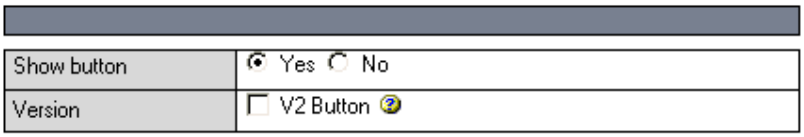
On each button you can specify:

Basics:



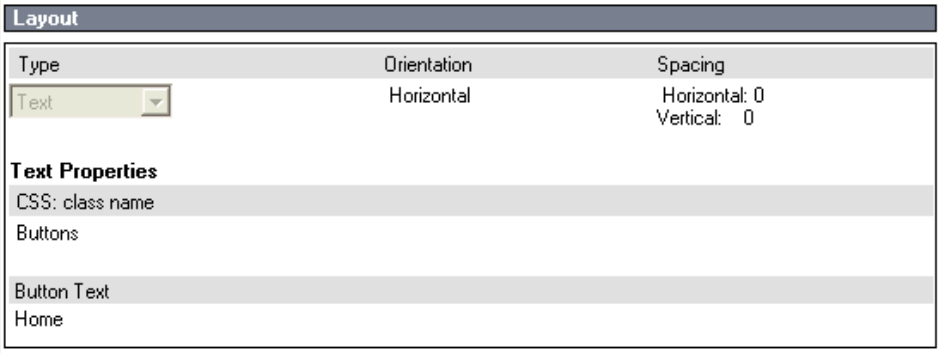
Cluster: The cluster containing the button. Clusters group buttons
Parent: It is possible to create a hierarchy from buttons for easy positioning
Position: The ordering of the button. Various sequences can be created by manipulating the values.

SHOW YES/NO:



Must this button be shown

LAYOUT

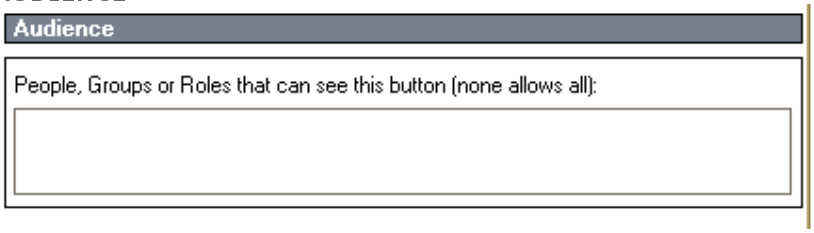


Type: the presentation of the button, the CSS to be used, if it is a graphic button or not etc. Your Passage recommends: Text buttons with CSS Positioning. This offers maximum flexibility.

Text properties: “CSS: class name”. Give the buttons a “class name” which can be addressed form a CSS style sheet.

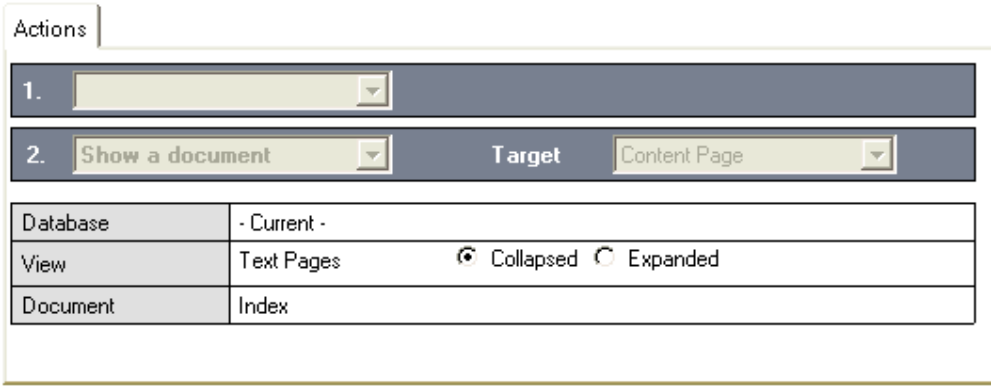
Button Text: the text on the button.

AUDIENCE



The audience option is part of the Intranet version of Passage. This option sets the readers field of the button. If the reader is in the specified group or role (listed in the Server's central Name and Address book), the button is visible from the Web. Otherwise the button is hidden. In this way the user is offered relevant information only.

ACTIONS



The action to be performed on the button. This is identical to the actions in the Frames.
Please note that if you use buttons to feed menu's, for example TIGRA menus, you need a javaScript view and connector. Only the first action is exported in the JavaScript view.

5.2 Controlling the layout of the button through CSS

It is possible to define special graphical effects on buttons without the creation of special GIF images for each state (for example mouse-over, mouse-down and steady state).

The major advantage of this technique is the speed (no downloads, rendering by the web-browser). A class is selected from the button document:

- Defining an CSS Style that governs the display of the buttons (for example .buttons)
- Creation of a CSS definition page in the Passage configuration menu
- Linking the CSS definition page onto the button navigator using the layout definition

Frequently used example:

- Define a graphics + text buttons, using a mouse- over effect (the image is a generic button image)

- The Cascading Style positions the text on top of the button, for example using the following css definition:

```
.Buttons
{ position: relative;
left:-100px;
top:-17px;
font-family:arial,verdana;
font-weight:bold;
font-size:10pt;
color:#1D3854
}
```

Where the text normally is positioned at the right-bottom of the image, the relative positioning places the text over the image. For more examples, please refer to the configuration examples on our website. **Even Better: Get a CSS book or browse the Web. CSS Examples are everywhere.**

5.3 Button import / export facility

This function no longer works in this version of Passage CMS.

6 Layout definition

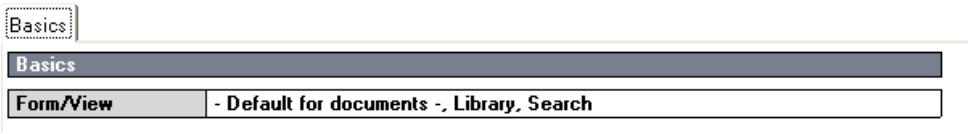
Passage uses Layout definitions to determine how various elements (navigators, documents and views) are presented.

Passage stores content and 'web presentation of this content' in separate locations. Content is stored in various document types, the lay-out is stored in the Configuration menu.
Passage Layout definitions are used to define the layout for the web-browser. When a navigator, document or view is accessed on the web, the layout definition is added to the document. Because content and presentation are stored independently it is easy to change them.

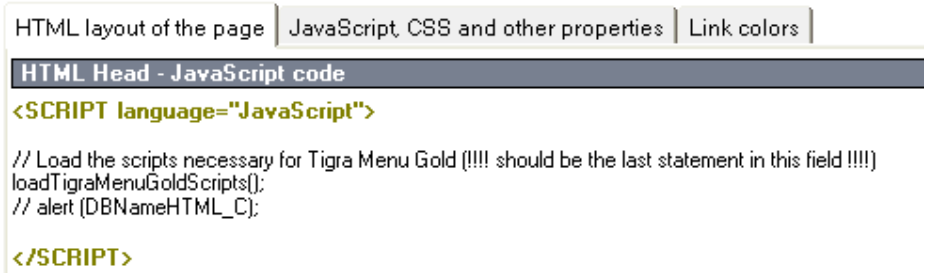
6.1 Generic Fields on the Layout Definition

Layout definitions are the Glue between the content, the navigation and the presentation. There are many ways to create layout definitions. Please be reluctant in building new layout definitions, there must be a smarter way to reach your goal then to maintain 100 layout definitions for a single site.

Identification and Bind (The name of the form or view to which the definition applies.)



HTML Layout of the Page (HTML Head)
It is possible to inject HTML code in the HTML Head of the Page.



HTML Layout of the Page (HTML before the title)
It is possible to inject HTML code in the HTML Page prior to the Title. In this example we use a Frameless site with DrempelsWeg (disability) implementation.

1. < HTML code *before* the title (available JavaScript functions)

```
<DIV id="crawlView"><BR><A HREF="/passage/homepage.nsf/v14013!openview">.</A></DIV>
<DIV id="TOP">
   <DIV id="accessibility-css">
      <SPAN id="plain" onclick="CSSCookies.fullreset()"><img src="/intraworks/internet/frameless.nsf/v9906/drempelsweg/$File/Font-small.gif" border=0 alt="Normaal letter formaat"></SPAN>
      <SPAN id="larger" onclick="CSSCookies.setLarger()"><img src="/intraworks/internet/frameless.nsf/v9906/drempelsweg/$File/Font-medium.gif" border=0 alt="Groter letter formaat"></SPAN>
      <SPAN id="largest" onclick="CSSCookies.setLargest()"><img src="/intraworks/internet/frameless.nsf/v9906/drempelsweg/$File/Font-large.gif" border=0 alt="Grootste letter formaat"></SPAN>
      <SPAN id="bw" onclick="CSSCookies.setBW()"><img src="/intraworks/internet/frameless.nsf/v9906/drempelsweg/$File/zetop.gif" border=0 alt="Verander kleur weergave"></SPAN>
   </DIV>
   <DIV id="searchBox">
<FORM name="searchform"><NOBR><INPUT name="Search" type="text" class="InputFld" value="Zoeken..." onFocus="this.value=''" onBlur="checkContent(this)" onKeyDown="checkEnter()" onCh
type="button" value="GAAN" class="InputBttn" style="height:20px;" onclick="openSearch(document.forms[0].Search.value)"></NOBR></FORM>     </DIV>
</DIV>
<DIV id="MIDDLE">
        <DIV id="MIDDLE-Navigation">
        <!-- Tigra Menu Gold, should be placed just before the </BODY> tag at the end of the document -->
                <SCRIPT language="JavaScript" type="text/javascript">
                        showTigraMenuGold( "MENU_ITEMS0", "MENU_SCOPE0", false );
                </SCRIPT>
                <!-- / Tigra Menu Gold -->
        </DIV>
</DIV>
<DIV id="BOTTOM">
<DIV id="BOTTOM_Content"><IMG align="left" height="600" width="1px" src="/intraworks/internet/frameless.nsf/v9906/IW/$File/Vertical.gif" border="0">
<DIV id="ToPrint">
<SPAN class="Bottom_Button_Back_Top" onclick="history.go(-1)"><IMG src="/intraworks/internet/frameless.nsf/v9906/IW/$File/Back.gif" border=0 alt="<< Back"></SPAN>
<SPAN class="Bottom_Button_Print_Top" onclick="openPrintWindow()"><IMG src="/intraworks/internet/frameless.nsf/v9906/IW/$File/Print.gif" border=0 alt="Print"></SPAN><BR><BR>
<DIV id="Contact_Links" class="Links"></DIV>
```

HTML Layout of the Page (HTML after the title)

After the title but before the notes rich text item containing all content

2. » HTML code *after* the title (available JavaScript functions)

```
<BR><BR>
```

HTML Layout of the Page (HTML at the end of the document)

Closing statements at the end of the page

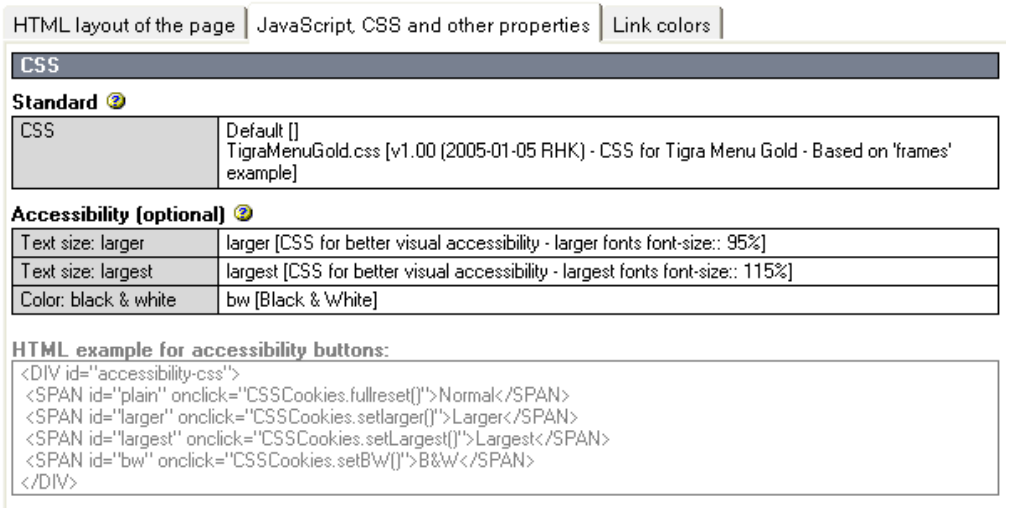
3. - HTML code at the end of the page (available JavaScript functions)

```
</DIV>
</DIV>

<SPAN class="Bottom_Button_Back" onclick="history.go(-1)"><IMG src="/intraworks/internet/frameless.nsf/v9906/IW/$File/Back.gif" border=0 alt="<< Back"></SPAN>
<SPAN id="CopyrightNotice" style="left:38%;"><CENTER>&copy; 1996 - 2005 Intraworks Solutions B.V.</CENTER></SPAN>
<SPAN class="Bottom_Button_Print" onclick="openPrintWindow()"><IMG src="/intraworks/internet/frameless.nsf/v9906/IW/$File/Print.gif" border=0 alt="Print"></SPAN><BR><BR>
</DIV>
```

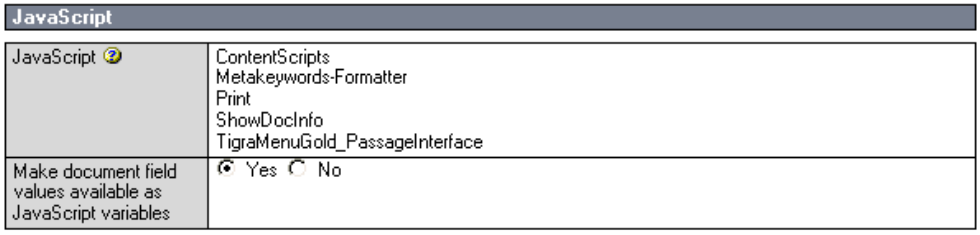
CSS Files

CSS Files to be bound to this page can be defined on this part. CSS Files are stored in the CSS Store in the Configuration.



JavaScript Files

JavaScript Files to be bound to this page can be defined on this part. JavaScript Files are stored in the JavaScript Store in the Configuration.



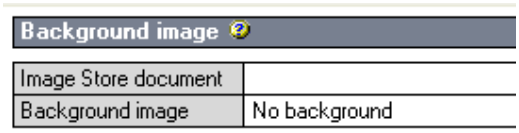
Document information is available in JavaScript

Specific document information can be made available to the end-user (i.e. browsing the website). This is done by optionally making the document info (or meta data) available to JavaScript. Via a specific function, this info can be placed everywhere on the screen. Information available is:

```
var title = "05.1 Buttons definition documents";
var summary = "";
var author = "Oscar Nijst";
var libraryCategory = "Manuals";
var keywords = "Passage Configuration-CMS 2.0";
var sKeywords_1 = "Passage Configuration";
var sKeywords_2 = "CMS 2.0";
var sKeywords_Meta_1 = "Manuals_Passage_Configuration_2_0";
var sKeywords_Meta_2 = "";
var source = "";
var dateRevision = "";
var dateCreated = "16-11-2005 14:40:56";
var dateModified = "16-11-2005 16:31:00";
var datePublished = "14-05-2002";
var sEmail = "";
var sRefID = "";
var sDocIcons = "";
```

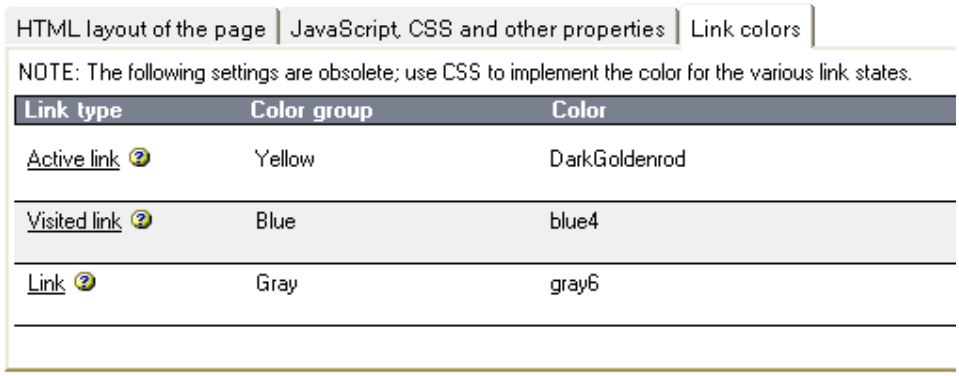
No longer actively used are:

Background images



You should specify a background image for the BODY element in your CSS definition instead.

Link colors



You should specify link colors for all elements in your CSS definition instead.

6.2 HTML layout of the page

As explained, Passage facilitates adding HTML to the presentation of the page on the web. This is done by injecting HTML on the Page.
Notes and Passage don't change the HTML in any way. In addition, already existing HTML code from existing layout definitions can be integrated in new definitions.

Passage documents are always presented in the browser using two fields, the **Title** and the **Body**. In the Layout Definition HTML code can be inserted in front of the title, between the title and the body and behind the body field. The Webserver generates the desired screen layout.

An example of how HTML code can be used to determine the layout of a document.

HTML Head
Allow pass-through html to be placed in the html-head part of the HTML document. This is necessary for specific JavaScript functions to work.
Also a generic variable called DBNameHTML (DBNameHTML_C in Master/Slave configurations) is made public on this page.

Necessary for:

- Dynamic HTML
- JavaScript Views
- Browser specific issues (by putting self modifying code in this field)

<SCRIPT language="JavaScript">

// Load the scripts necessary for Tigra Menu Gold (!!!! should be the last statement in this field !!!!)
loadTigraMenuGoldScripts();
// alert (DBNameHTML_C);

</SCRIPT>

Before the title

This field can be used to define a table with three columns, for example, placing the next and previous buttons in the first cell, and to defining the layout of the title. The roll down menu (click available JavaScript functions) shows the available JavaScript functions .

```
<DIV id="crawlView"><BR><A
HREF="/passage/homepage.nsf/v14013!openview">.</A></DIV>
<DIV id="TOP">
<DIV id="accessibility-css">
<SPAN id="plain" onclick="CSSCookies.fullreset()"><img
src="/intraworks/internet/frameless.nsf/v9906/drempelsweg/$File/Font-
small.gif" border=0 alt="Normaal letter formaat"></SPAN>
<SPAN id="larger" onclick="CSSCookies.setLarger()"><img
src="/intraworks/internet/frameless.nsf/v9906/drempelsweg/$File/Font-
medium.gif" border=0 alt="Groter letter formaat"></SPAN>
<SPAN id="largest" onclick="CSSCookies.setLargest()"><img
src="/intraworks/internet/frameless.nsf/v9906/drempelsweg/$File/Font-
large.gif" border=0 alt="Grootste letter formaat"></SPAN>
<SPAN id="bw" onclick="CSSCookies.setBW()"><img
src="/intraworks/internet/frameless.nsf/v9906/drempelsweg/$File/zetop.gif"
border=0 alt="Verander kleur weergave"></SPAN>
</DIV>
<DIV id="searchBox">
<FORM name="searchform"><NOBR><INPUT name="Search" type="text"
class="InputFld" value="Zoeken..." onFocus="this.value=''"
onBlur="checkContent(this)" onKeyDown="checkEnter()"
onChange="openSearch(this.value)"> <INPUT type="button" value="GAAN"
class="InputBttn" style="height:20px;"
onclick="openSearch(document.forms[0].Search.value)"></NOBR></FORM>
```

```
</DIV>
</DIV>
<DIV id="MIDDLE">
<DIV id="MIDDLE-Navigation">
<!-- Tigra Menu Gold, should be placed just before the </BODY> tag at the
end of the document -->
<SCRIPT language="JavaScript" type="text/javascript">
showTigraMenuGold( "MENU_ITEMS0", "MENU_SCOPE0", false );
</SCRIPT>
<!-- / Tigra Menu Gold -->
</DIV>
</DIV>
<DIV id="BOTTOM">
<DIV id="BOTTOM_Content"><IMG align="left" height="600" width="1px"
src="/intraworks/internet/frameless.nsf/v9906/IW/$File/Vertical.gif"
border="0">
<DIV id="ToPrint">
<SPAN class="Bottom_Button_Back_Top" onclick="history.go(-1)"><IMG
src="/intraworks/internet/frameless.nsf/v9906/IW/$File/Back.gif" border=0
alt="<< Back"></SPAN>
<SPAN class="Bottom_Button_Print_Top"
onclick="openPrintWindow()"><IMG
src="/intraworks/internet/frameless.nsf/v9906/IW/$File/Print.gif" border=0
alt="Print"></SPAN><BR><BR>
<DIV id="Contact_Links" class="Links"></DIV>
```

After the title [IMAGE] This option is often used to go to a new row/cell of a table.

```
<BR><BR>
```

After the end ot the Page

```
</DIV>
</DIV>
<SPAN class="Bottom_Button_Back" onclick="history.go(-1)"><IMG
src="/intraworks/internet/frameless.nsf/v9906/IW/$File/Back.gif" border=0
alt="<< Back"></SPAN>
<SPAN id="CopyrightNotice" style="left:38%;"><CENTER>&copy; 1996 -
2005 Intraworks Solutions B.V.</CENTER></SPAN>
<SPAN class="Bottom_Button_Print" onclick="openPrintWindow()"><IMG
src="/intraworks/internet/frameless.nsf/v9906/IW/$File/Print.gif" border=0
alt="Print"></SPAN><BR><BR>
</DIV>
```

6.3 Determining which Layout Definition is used by a web page

When debugging a Passage website, it is possible to determine which layout definition is used. Follow these steps:

1. Load the page in the browser
2. View the HTML source
3. Search for the text “Layout Definition info” or “Lookup Key”

In the HTML source, you’ll find something like this:

```
<!--
Layout Definition info
LookupDB: /passage/personal/tainsh.nsf
LookupKey: Text PageTEST
For db: non-db specific
-->
```

Where *layoutdefinition* stands for the used layout definition document

6.4 Changing the layout bay passing a URL parameter

It is possible to open a document with a different layout definition then configured. Create a link or button with a url to the desired document. Then add the code “&lookupkey=<keyOfThe LayoutDefiniton>” to the url. Resulting in opening the desired document but with the layout definition passed in the URL.

Keep the following in mind:

1. Passing a key in a url overrules the predefined layout definition for the document
2. If the in the url passed layout definition does not exist, the document will be opened with a layout definition, it does not fall back on the predefined layout definition.

Example:
http://paco/Develop/Interactivity/im3.42dev.nsf/CreateDoc?OpenAgent&Type=ResponseForm&lookupkey=KeyOfffTheLayoutDefinition

7 Context Definitions

7.1 General information

A Context Definition enables documents to incorporate dynamic content, such as navigation, views and relevant documents, that will be 'injected' into placeholders present on the Layout Definition.

The various types of dynamic content will be referred to as "Context Objects".

7.1.1 How it works

Each time a page is requested from the browser, the "PageBuilder" agent is run. This agent will try to look up the Context Definition for the document – either the one specified on the document, or the default one (if existent). If it has found this Context Definition, the PageBuilder will initialise all the Context Objects on the definition that have been enabled, and will try to find special placeholders on the Layout Definition that's loaded, to substitute those with the HTML generated by the Context Objects.

This means that only enabling a Context Object is not enough to actually show the object on the web page; it is also necessary to put a placeholder on the Layout Definition that is used to display the document, otherwise the object will be initialised, but there is nothing to write the results to.

Below is an example of four placeholders on a Layout Definition:

1. « HTML code before the title (available JavaScript functions)

```
<DIV id="ctxNavigation1">%ctxNavigation1%</DIV>
<DIV id="ctxNavigation2">%ctxNavigation2%</DIV>
<DIV id="ctxNavigation3">%ctxNavigation3%</DIV>
<DIV id="ctxNavigation4">%ctxNavigation4%</DIV>
```

The placeholders in this example are the string surrounded by the percent signs: %.....%. These will be substituted completely by the HTML-code from the various Context Objects.

If a placeholder has been specified on the Layout Definition, but there's no matching Context Object on the Context Definition, the PageBuilder will replace the entire placeholder with an empty string. This means it's possible to include many placeholders on a Layout Definition, even though not every placeholder will be actually used by the Context Definition.

In this example, the id's of the DIV tags are the same as the placeholder names. Although not mandatory, it makes exchanging CSS style sheets that describe the looks of the objects much easier.

7.1.2 Determining which Context Definition is used by a page

To help determining which Context Definition is used by the Passage CMS web page that's opened in a browser, the source of each page contains information about this. Open the HTML source of the document, and you'll find the following string near the top of the document.

If Context Definition "Default" is in use, you'll see the following:

<!-- Context Definition="Default" -->

If **no** Context Definition is being used, you'll see the following in the HTML source:

<!-- Context Definition="No Context Definition found" -->

There are only two situations in which you'll see the message "No Context Definition found":

1. No Context Definition has been specified on the document, and Context Definition "Default" doesn't exist

2. A Context Definition has been specified on the document, but it doesn't exist.

Sites that don't use Context Definitions will always have "No Context Definition found" in the HTML source code.

7.1.3 Mandatory fields and @Formula fields

Mandatory fields have their field description in **bold type**. These fields are only mandatory if the Context Object which the field is in, is enabled (checked field "Placeholder name").

Fields that can also take @Formulas as field values have a red asterisk * next to the description. Field values starting with an "@"-sign, will be evaluated as an @Formula. That first "@"-sign itself will be ignored.

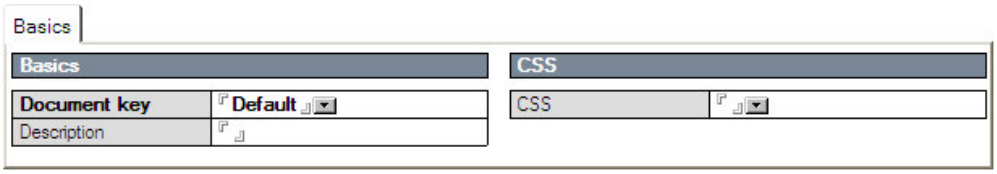
7.2 Basic Configuration of a Context Definition

7.2.1 Fields present within each Context Object

Field "**Placeholder**" has a unique value for each Context Object. If a Context Object has been enabled by checking the checkbox, the PageBuilder will initialise the object when a page is requested, and will try to find the appropriate placeholder on the Layout Definition to replace it by the generated HTML-code.

The fields "**Script Library**" and "**Class name**" are a pair. They can be used to override the 'standard behaviour' of each of the Context Objects, so they can be implemented in a customer specific way.

7.2.2 Table block – Basics



7.2.2.1 Field - Document Key

This is the unique name of the Context Definition. There is **no** checking whether this name is unique or not.

To make a Context Definition that will be used for all documents, use "Default" as a Document Key.

The contents of this field is visible to Editors when they choose a specific Context Definition on a document.

7.2.2.2 Field - Description

Use this field to give a detailed description of the functionality that you have configured on the Context Definition. The contents of this field are visible to Editors when they choose a specific Context Definition on a document.

Always make sure you give your Context Definitions an accurate description of what they do, because after a while chances are that it's not so clear anymore what each one does if the description is too short.

7.2.3 Table block - CSS

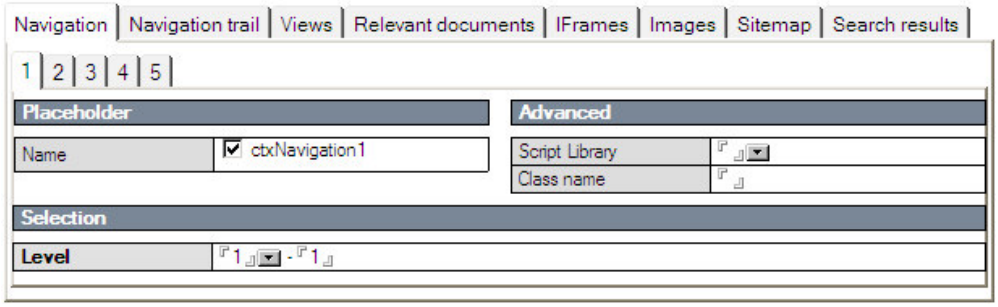
7.2.3.1 Field - CSS document

This field allows you to load an additional CSS style sheet for documents that use this Context Definition, should the need arise.

The CSS style sheet specified here will be loaded after the style sheets from the Layout Definition have been loaded.

7.3 Context Object “Navigation” 1 - 5

Allows you to incorporate up to 5 Navigation elements on a page.



Default behaviour: single-level navigation

The standard Script Library and Class will show all documents on **one** level of the navigation path only. It uses the “Document as Button” fields from the current document to determine what that path is. If a document hasn’t specified something in the “Document as Button”-fields, no navigation will be shown for that document.

The styling of the output of objects of this type is entirely done by CSS.

Below is an example of HTML output for a Navigation context object:

```
<DIV id="ctxNavigation1">
<UL>
<LI class="first on"><DIV><A
href="/passage/personal/tainsh.nsf/0/41204EF88292153AC12571B200287D9F?OpenDo
cument">Homepage</A></DIV></LI>
<LI class=""><DIV><A
href="/passage/personal/tainsh.nsf/0/1C1D5830E1624872C12571B200352D94?OpenDo
cument">News overview</A></DIV></LI>
<LI class="last"><DIV><A
href="/passage/personal/tainsh.nsf/0/B7C76A0EFFD7289CC12571B600538DEB?OpenD
ocument">Products overview</A></DIV></LI>
</UL>
</DIV>
```

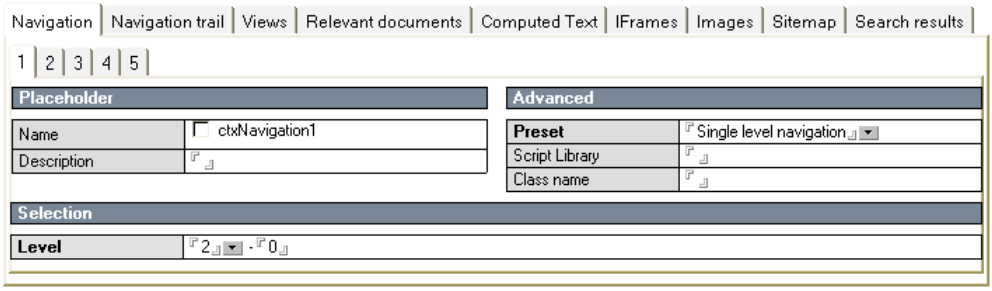
Combined with a CSS style sheet, this will be rendered like this in the browser:

Alternative behaviour: tree-style navigation (multi-level)

Some sites may require a navigation that is also known as “tree-style” navigation. An example of this is the way folders are shown in the Windows Explorer interface.

Example “tree style” menu

To use this type of navigation, choose “sl.iw.iwPassageCMS.CTXNavigationMulti” from the pull-down list in the “Script Library” field. The field “Class name” will automatically be filled out with the pre-defined matching Class:



NB: It is possible to use this alternative even for navigation that only needs to show one level (fields “Level from” and “Level to” set to the same number); however, this script is slower than the default one that is specially tailored to that purpose.

7.3.1 Table block - Placeholder

7.3.1.1 Field - Name

When checked, this Context Object will be used when generating the web page.

7.3.1.2 Field - Description

Optional. A description of the view navigation on this tab.

7.3.2 Table block - Advanced

7.3.2.1 Field - Script Library

The standard LotusScript library that is used if this field is left empty is:

sl.iw.iwPassageCMS.CTXNavigation

The following value can be chosen from the pull-down list:

sl.iw.iwPassageCMS.CTXNavigationMulti

7.3.2.2 Field - Class name

The standard Class that is used if this field is left empty is:

Public Class clCTXNavigation As clCTXBaseView

The value for this field will be automatically set to "*clCTXNavMulti*" if field "Script Library" has "*sl.iw.iwPassageCMS.CTXNavigationMulti*" as value.

7.3.3 Table block - Selection

7.3.3.1 Field - Level from

This field specifies from which level the navigation should be retrieved. Level 1 is the lowest level possible.

For example:

- if you specify “1” in this field, all documents/buttons that have the same navigation path as the current document up till level 1 will be included in the navigation.
- If you specify “3” in this field, the documents/buttons that have a matching navigation path up till level 3 will be included.

The default value for each ctxNavigation object is the same as its’ object number.

7.3.3.2 Field - Level to

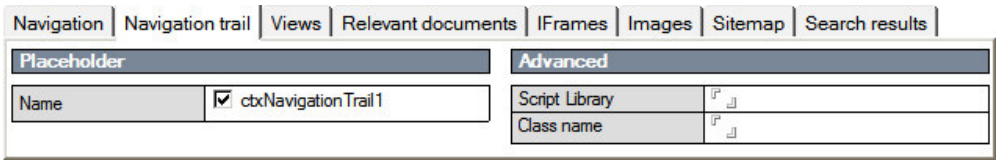
NB: this field is only used if field “Script Library” is not empty, for example when the “tree-style” navigation is desired.

It defines up till which level the documents/buttons should be included in the navigation. Level 1 is the lowest; theoretically there is no highest level. To make sure all levels are included, choose “End” from the list, or type “0”.

The default value for each ctxNavigation object is the same as its’ object number.

7.4 Context Object “Navigation Trail”

Enables you to show a navigation trail on your documents, so users can see a clickable ‘trail’ representing the navigational hierarchy in which the document is located.



An example how this might look in a browser:

You are here: Home > News overview > Product announcements > New alloy discovered

7.4.1 Table block - Placeholder

7.4.1.1 Field - Name

When checked, this Context Object will be used when generating the web page.

7.4.1.2 Field - Description

Optional. A description of the navigation trail defined on this tab.

7.4.2 Table block - Advanced

7.4.2.1 Field - Script Library

The standard LotusScript library that is used if this field is left empty is:

sl.iw.iwPassageCMS.CTXNavigationTrail

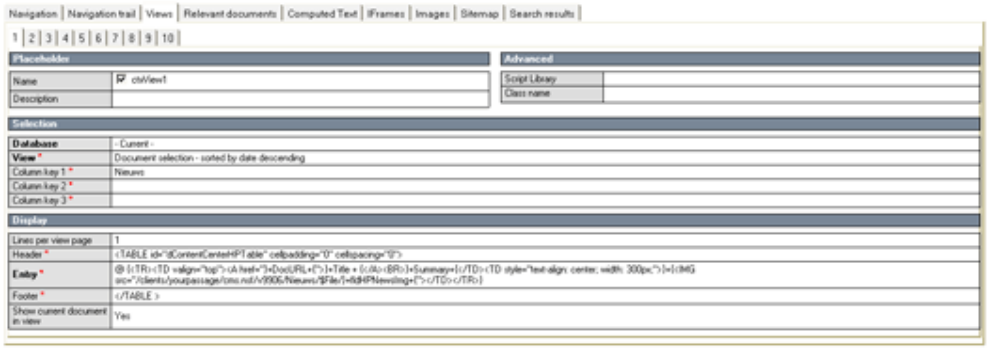
7.4.2.2 Field - Class name

The standard Class that is used if this field is left empty is:

Public Class clCTXNavigationTrail

7.5 Context Object “View” 1 - 10

Enables you to include up to 10 views on a page.



7.5.1 Table block - Placeholder

7.5.1.1 Field - Name

When checked, this Context Object will be used when generating the web page.

7.5.1.2 Field - Description

Optional. A description of the view defined on this tab.

7.5.2 Table block - Advanced

7.5.2.1 Field - Script Library

The standard LotusScript library that is used if this field is left empty is:

sl.iw.iwPassageCMS.CTXView

7.5.2.2 Field - Class name

The standard Class that is used if this field is left empty is:

Public Class clCTXView As clCTXBaseView

7.5.3 Table block - Selection

7.5.3.1 Field - Database

Specifies the database to get the view from.

The choices shown are from the Database Administration field “Passage CMS Slave databases".

7.5.3.2 Field - View

The name or alias of the view that you want to show. The list of choices is a selection of views that are present in every Passage CMS database. It is however also possible to use any view name or alias here. It is also an option to use @Formula, e.g. for retrieving the view name from an URL parameter.

An @Formula can be used if the first character is an “@”-sign.

7.5.3.3 Field - Column key 1

When used, restricts the documents to be shown to only those that have this value in the first view column. This only works if the values in that column are of type “string”.

An @Formula can be used if the first character is an “@”-sign.

7.5.3.4 Field - Column key 2

When used, restricts the documents to be shown to only those that have this value in the second view column. This only works if the values in that column are of type “string”.

An @Formula can be used if the first character is an “@”-sign.

7.5.3.5 Field - Column key 3

When used, restricts the documents to be shown to only those that have this value in the third view column. This only works if the values in that column are of type “string”.

An @Formula can be used if the first character is an “@”-sign.

7.5.4 Table block - Display

7.5.4.1 Field - Lines per view page

Can be used to restrict the number of documents shown.

Leave this field empty or specify 0 to show all documents.

7.5.4.2 Field - Header

HTML code that will be shown above the actual view, for example an opening <DIV> tag.

An @Formula can be used if the first character is an “@”-sign.

All fields from the current document are available, as well as the following variables:

CollectionCount	Number, the total number of documents in the view
Maxlines	Number, the number of documents per page
ViewStart	Number, the counter from which the documents are shown
ViewEnd	Number, the counter to which the documents are shown
ViewNavigation	String, buttons for navigating view pages

7.5.4.3 Field - Entry

HTML code that is used for every entry in the view.

An @Formula can be used if the first character is an “@”-sign; in this field, this will always be the case.

All fields from the document being processed are available, as well as the following variables:

CurrDocUNID	String, UNID of the current document (handle to the document that is opened in the browser, not to the document in the collection)
DocURL	String, URL of the document
DBNameHTML	String, Master database path
DBNameHTML_C	String, Current database path
DocNumber	String, number in the view
OddEven	String, "Odd" or "Even"

You can use the following @Formula to show a view entry:

@{<DIV><A href="} + DocURL + {">} + Title + {</A></DIV>}

7.5.4.4 Field - Footer

HTML code that will be shown below the actual view, for example a closing </DIV> tag.

An @Formula can be used if the first character is an “@”-sign.

All fields from the current document are available, as well as the following variables:

CollectionCount	Number, the total number of documents in the view
Maxlines	Number, the number of documents per page
ViewStart	Number, the counter from which the documents are shown
ViewEnd	Number, the counter to which the documents are shown
ViewNavigation	String, buttons for navigating view pages

7.5.4.5 Field - Show current document in view

Determines wether the current document should also be shown in the view, should it also occur that it is listed in that view.

7.5.5 Available URL parameters

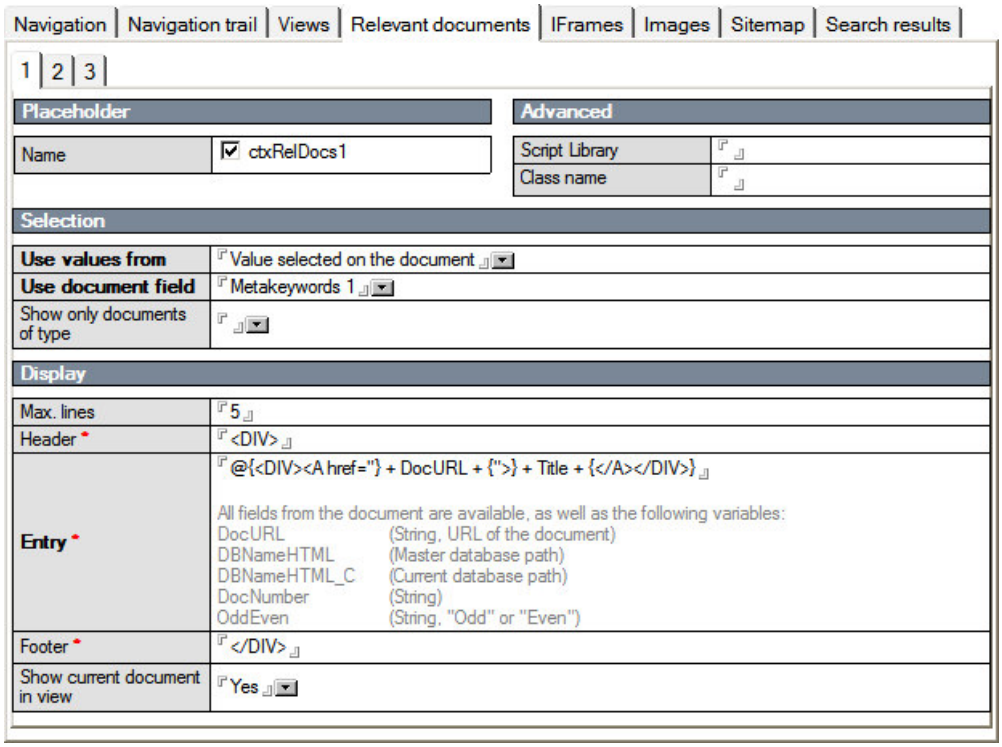
The following URL parameters can be used for this type of Context Object:

count	Amount of documents to be shown. Overwrites any value specified in field “Lines per view page”.
start	View row from which the view contents should be shown.

Please refer to “7.13 Passing parameters to Context Objects” for more information about passing URL parameters to specific Context Objects.

7.6 Context Object “Relevant documents” 1 - 3

Enables you to include a maximum of 3 lists of documents that are relevant to the current document, based on various criteria.



7.6.1 Table block - Placeholder

7.6.1.1 Field - Name

When checked, this Context Object will be used when generating the web page.

7.6.1.2 Field - Description

Optional. A description of the relevant documents defined on this tab.

7.6.2 Table block - Advanced

7.6.2.1 Field - Script Library

The standard LotusScript library that is used if this field is left empty is:

sl.iw.iwPassageCMS.CTXRelDocs

7.6.2.2 Field - Class name

The standard Class that is used if this field is left empty is:

Public Class clCTXRelDocs As clCTXBaseView

7.6.3 Table block - Selection

7.6.3.1 Field - Use values from

To use this Context Object, it is necessary to specify what the criteria are that should be met to include other documents that are 'relevant' to the current document.

This field provides two options from which to take the values:

1. The values to be used will be chosen on this Context Definition:

Selection	
Use values from	This Context Definition
Metakeyword list	A
Metakeyword	Groep 1
Show only documents of type	

   Specify the "Metakeyword list" and the "Metakeyword" that will be used to look up relevant documents. All documents that have the same values, will be included in the view.

2. The values to be used will be on the current document:

Selection	
Use values from	Value selected on the document
Use document field	Metakeywords 1
Show only documents of type	

   Specify which document field (Metakeywords 1 or Metakeywords 2) should be used.

7.6.4 Table block - Display

7.6.4.1 Field - Max. lines

Can be used to restrict the number of documents shown.

Leave this field empty or specify 0 to show all documents.

7.6.4.2 Field - Header

HTML code that will be shown above the actual view, for example an opening <DIV> tag.

An @Formula can be used if the first character is an “@”-sign.

All fields from the current document are available, as well as the following variables:

CollectionCount	Number, the total number of documents in the view
Maxlines	Number, the number of documents per page
ViewStart	Number, the counter from which the documents are shown
ViewEnd	Number, the counter to which the documents are shown

7.6.4.3 Field - Entry

HTML code that is used for every entry in the view.

An @Formula can be used if the first character is an “@”-sign; in this field, this will always be the case.

All fields from the document being processed are available, as well as the following variables:

DocURL	String, URL of the document
DBNameHTML	Master database path
DBNameHTML_C	Current database path
DocNumber	String
OddEven	String, "Odd" or "Even"

You can use the following @Formula to show a view entry:

@{<DIV><A href="} + DocURL + {">} + Title + {</A></DIV>}

7.6.4.4 Field - Footer

HTML code that will be shown below the actual view, for example a closing </DIV> tag.

An @Formula can be used if the first character is an “@”-sign.

All fields from the current document are available, as well as the following variables:

CollectionCount	Number, the total number of documents in the view
Maxlines	Number, the number of documents per page
ViewStart	Number, the counter from which the documents are shown
ViewEnd	Number, the counter to which the documents are shown

7.6.4.5 Field - Show current document in view

Determines wether the current document should also be shown in the view, should it also occur that it is listed in that view.

7.6.5 Available URL parameters

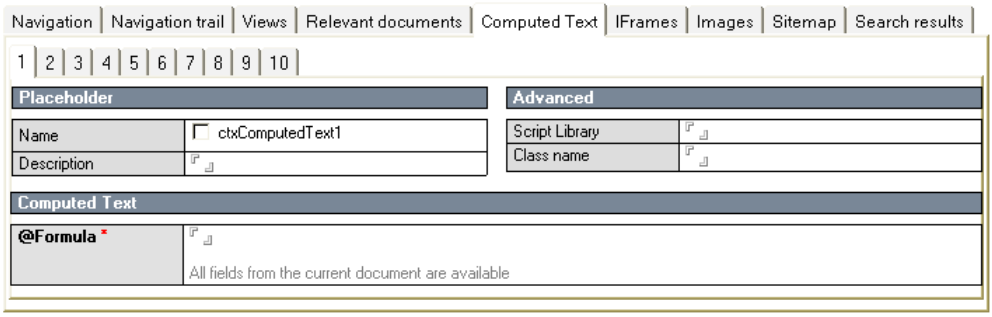
The following URL parameters can be used for this type of Context Object:

count	Amount of documents to be shown. Overwrites any value specified in field “Lines per view page”.
start	View row from which the view contents should be shown.

Please refer to “7.13 Passing parameters to Context Objects” for more information about passing URL parameters to specific Context Objects.

7.7 Context Object “Computed Text” 1 - 5

Enables you to include a maximum of 5 blocks of HTML/JavaScript/Text/etc.



7.7.1 Table block - Placeholder

7.7.1.1 Field - Name

When checked, this Context Object will be used when generating the web page.

7.7.1.2 Field - Description

Optional. A description of the Computed Text defined on this tab.

7.7.2 Table block - Advanced

7.7.2.1 Field - Script Library

The standard LotusScript library that is used if this field is left empty is:

sl.iw.iwPassageCMS.CTXIFrame

7.7.2.2 Field - Class name

The standard Class that is used if this field is left empty is:

Public Class clCTXIFrame

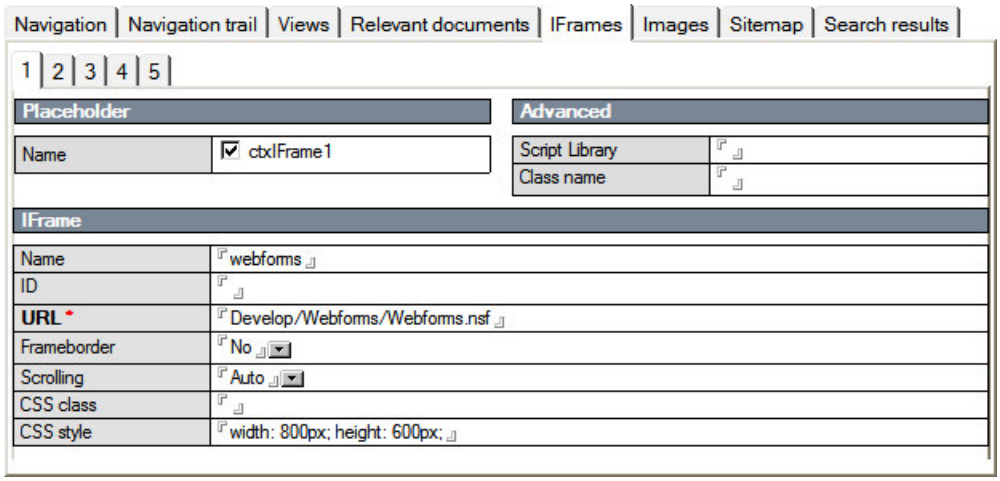
7.7.3 Table block - Computed Text

*7.7.3.1 Field - @Formula**

Specify the value for you want placed in the named placeholder. It is possible to use @Formula.

7.8 Context Object “IFrame” 1 - 5

Enables you to include a maximum of 5 IFrames on your documents.



7.8.1 Table block - Placeholder

7.8.1.1 Field - Name

When checked, this Context Object will be used when generating the web page.

7.8.2 Table block - Advanced

7.8.2.1 Field - Script Library

The standard LotusScript library that is used if this field is left empty is:

sl.iw.iwPassageCMS.CTXIFrame

7.8.2.2 Field - Class name

The standard Class that is used if this field is left empty is:

Public Class clCTXIFrame

7.8.3 Table block - IFrame

7.8.3.1 Field - Name

Specify the value for HTML attribute "NAME" here. This is only useful if you need to address the IFrame from a script.

7.8.3.2 Field - Description

Optional. A description of the iframe defined on this tab.

7.8.3.3 Field - ID

Specify the value for HTML attribute "ID" here. This is only useful if you need to address the IFrame from a script.

7.8.3.4 Field - URL

Specify the URL for the IFrame here.

An @Formula can be used if the first character is an “@”-sign.

7.8.3.5 Field - Frameborder

This is a standard HTML property of an IFrame, and included here for completeness.

In most cases setting this to “No” is best, especially because it’s possible to style the border using CSS style sheets too, in a more precise way.

7.8.3.6 Field - Scrolling

Defines wether scroll bars should appear within the IFrame.

Auto
Scrollbars will only appear if the content of the IFrame exceeds the dimensions of the IFrame:
- if the content is too wide, a side scrollbar will be shown
- if the content is too high, a bottom scrollbar will be shown

Yes
Scrollbars will always appear.

No
Scrollbars will never appear.

7.8.3.7 Field - CSS class

The CSS class for the IFrame.

7.8.3.8 Field - CSS style

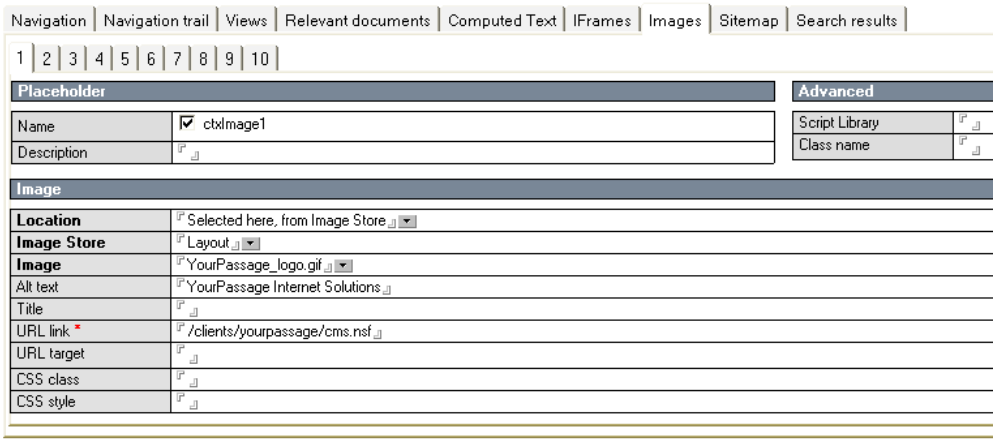
The inline CSS style for the IFrame.

This can be used to specify the dimensions and other aspects of the IFrame, for example:

width: 800px; height: 600px;

7.9 Context Object “Image” 1 - 10

Enables you to include up to 10 images on the page.



7.9.1 Table block - Placeholder

7.9.1.1 Field - Name

When checked, this Context Object will be used when generating the web page.

7.9.1.2 Field - Description

Optional. A description of the image defined on this tab.

7.9.2 Table block - Advanced

7.9.2.1 Field - Script Library

The standard LotusScript library that is used if this field is left empty is:

sl.iw.iwPassageCMS.CTXImage

7.9.2.2 Field - Class name

The standard Class that is used if this field is left empty is:

Public Class clCTXImage

7.9.3 Table block - Image

7.9.3.1 Field - Location

The URL of the image can be either selected on the Context Definition itself, or be specified by an @Formula. The latter enables you to use document values for the URL.

This field provides two options from which to take the values:

1. The image URL will be selected here, from the Image Store documents:

Image	
Location	Selected here, from Image Store
Image Store	Passage_Icons
Image	Multiculturalfood_Distribution.jpg
Alt text	Distribution of multi-cultural foods
Title	Distribution of multi-cultural foods

2. The image URL will be composed by using an @Formula:

Image	
Location	Get image URL from @Formula
URL *	

7.9.3.2 Field - URL link

Specify an optional URL link for the image, making the image clickable.

An @Formula can be used if the first character is an “@”-sign.

7.9.3.3 Field - URL target

Specify an optional URL target for link on the image.

7.9.3.4 Field - CSS class

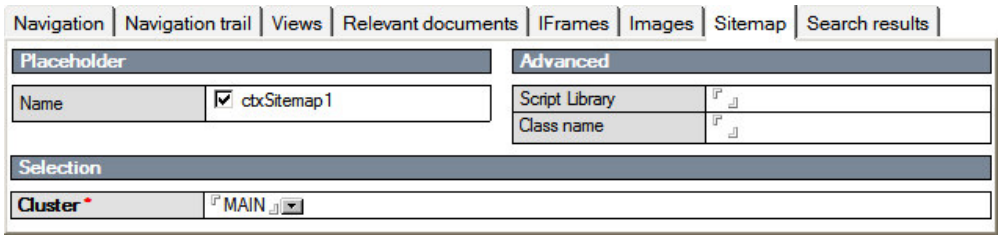
The CSS class for the image.

7.9.3.5 Field - CSS style

The inline CSS style for the image.

7.10 Context Object “Sitemap”

Enables you to incorporate a Sitemap on your page, showing all documents/buttons in the site in a hierarchical way, including links to each document.



The Sitemap is generated using <UL> and <LI> tags and styled exclusively by CSS style sheet definitions, of which an example is included in the “Default” style sheet present in the Passage CMS template.

A typical implementation of a Sitemap would be:

- Create a Context Definition with Key “Sitemap”
- Enable the Placeholder for Context Object “Sitemap”
- Create a Text Page with Key “Sitemap”
- On that Text Page, set the value of field “Context Definition” to “Sitemap”
- To open the Sitemap, create a link to the Text Page, e.g.: /dbpath/dbname.nsf/v01013/Sitemap?OpenDocument

7.10.1 Table block - Placeholder

7.10.1.1 Field - Name

When checked, this Context Object will be used when generating the web page.

7.10.1.2 Field - Description

Optional. A description of the sitemap defined on this tab.

7.10.2 Table block - Advanced

7.10.2.1 Field - Script Library

The standard LotusScript library that is used if this field is left empty is:

sl.iw.iwPassageCMS.CTXSitemap

7.10.2.2 Field - Class name

The standard Class that is used if this field is left empty is:

Public Class clCTXSitemap

7.10.3 Table block - Selection

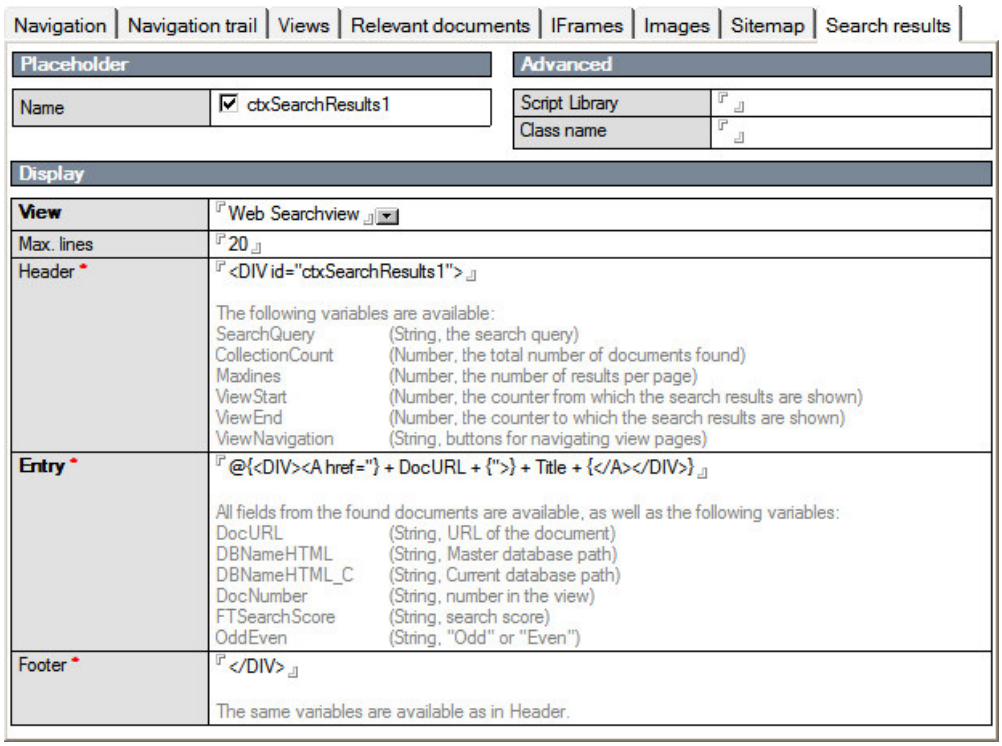
7.10.3.1 Field - Cluster

Specify which Cluster should be used to build the Sitemap.

An @Formula can be used if the first character is an “@”-sign.

7.11 Context Object “Search results”

Enables you to show search results on a page, based on parameters passed in by the URL.



7.11.1 Table block - Placeholder

7.11.1.1 Field - Name

When checked, this Context Object will be used when generating the web page.

7.11.1.2 Field - Description

Optional. A description of the searchresults defined on this tab.

7.11.2 Table block - Advanced

7.11.2.1 Field - Script Library

The standard LotusScript library that is used if this field is left empty is:

sl.iw.iwPassageCMS.CTXSearchResults

7.11.2.2 Field - Class name

The standard Class that is used if this field is left empty is:

Public Class clCTXSearchResults As clCTXBaseView

7.11.3 Table block - Display

7.11.3.1 Field - View

Defines which view will be used to perform the search on. Default is "Web Searchview".

7.11.3.2 Field - Max. lines

Can be used to restrict the number of search results that are returned.

Leave this field empty or specify 0 to return all search results.

7.11.3.3 Field - Header

HTML code that will be shown above the actual view, for example an opening <DIV> tag.

An @Formula can be used if the first character is an “@”-sign.

All fields from the current document are available, as well as the following variables:

SearchQuery	String, the search query
CollectionCount	Number, the total number of documents found
Maxlines	Number, the number of results per page
ViewStart	Number, the counter from which the search results are shown
ViewEnd	Number, the counter to which the search results are shown
ViewNavigation	String, buttons for navigating view pages

7.11.3.4 Field - Entry

HTML code that is used for every entry in the view.

An @Formula can be used if the first character is an “@”-sign; in this field, this will always be the case.

All fields from the document being processed are available, as well as the following variables:

DocURL	String, URL of the document
DBNameHTML	String, Master database path
DBNameHTML_C	String, Current database path
DocNumber	String, number in the view
FTSearchScore	String, search score
OddEven	String, "Odd" or "Even"

You can use the following @Formula to show a view entry:

@{<DIV><A href="} + DocURL + {">} + Title + {</A></DIV>}

7.11.3.5 Field - Footer

HTML code that will be shown below the actual view, for example a closing </DIV> tag.

An @Formula can be used if the first character is an “@”-sign.

All fields from the current document are available, as well as the following variables:

SearchQuery	String, the search query
CollectionCount	Number, the total number of documents found
Maxlines	Number, the number of results per page
ViewStart	Number, the counter from which the search results are shown
ViewEnd	Number, the counter to which the search results are shown
ViewNavigation	String, buttons for navigating view pages

7.11.4 Available URL parameters

The following URL parameters can be used for this type of Context Object:

count	Amount of documents to be shown. Overwrites any value specified in field “Lines per view page”.
start	View row from which the view contents should be shown.
query	The search query

An example of a URL that performs a search is:

http://paco/Develop/Passage/Passage3.0DEV.nsf/0/7660D335A9CFEA74C125714200483156?OpenDocument&ctxSearchResults1=(start=1~count=4~query=cms%20systems)

This URL will perform a search for “cms systems” and will show the results from document 1 up till the 4th document.

Please refer to “7.13 Passing parameters to Context Objects” for more information about passing URL parameters to specific Context Objects.

7.12 Table block - Overview of used objects

Shows which Context Objects are in use, along with their corresponding placeholders.

Overview of used objects	
Object name	**Placeholder name**
- Navigation 1	ctxNavigation1
- Navigation 2	ctxNavigation2
- Navigation 3	ctxNavigation3
- Navigation 4	ctxNavigation4
- Navigation trail	ctxNavigationTrail1
- View 1	ctxView1
- View 5	ctxView5
- View 6	ctxView6
- View 9	ctxView9
- Relevant documents 1	ctxRelDocs1
- Relevant documents 2	ctxRelDocs2

7.13 Passing parameters to Context Objects

It's possible to pass parameters to specific Context Objects through the URL with which the page is opened.

The format for these parameters is as follows:

&ctxObject=(param1=value1~param2=value2)

ctxObject	placeholder name of the Context Object
param1	name of the 1st parameter
value1	value for the 1st parameter
param2	name of the 2nd parameter
value2	value for the 2nd parameter

Example - a search query URL:

http://paco/Develop/Passage/Passage3.0DEV.nsf/0/7660D335A9CFEA74C125714200483156?OpenDocument&ctxSearchResults1=(start=1~count=4~query=cms%20systems)

This URL will do the following:

- search for query "cms systems"
- show the results from the 1st document
- show the results up to and including the 4th document.

Not all Context Object types have parameters. Refer to each specific Context Object for parameter listings.

7.14 Overriding the default behaviour of Context Objects

The functionality of each Context Object is coded in it's own LotusScript Library, in an Object Oriented way – each object has it's own class. These LotusScript Libraries are copyrighted by Your Passage; they are an integral part of Passage CMS en may not be altered in any way.

It is however possible to override their behaviour, by creating your own LotusScript Library which contains your own Class-definition, that overrides methods of the standard ones.

The field "Script Library" can be used to specify the LotusScript Library that should be used instead of the standard script library. Use field "Class name" to specify the name of the custom class within that script library.

To make sure that this custom class complies to the requirements of the Context Object, it should always use the script library that the standard class was in, and the Class definition should always be declared as being of the class type of that original, for example:

Options

Use "sl.iw.iwPassageCMS.CTXImage"

Declarations

Private Const CURRENT_MODULE="sl.iw.iwPassageCMS.CTXImageCustom"

Public Class clCTXImageCustom As clCTXImage

Inside this class, the method that is always used to return the HTML code for the Context Object is:

Public Function getHTML( ) As String

In most cases, this will be the only method that you will override.

8 Web front end formatting

8.1 JavaScript Library

The JavaScript library is used to store generic JavaScript routines. The advantage of this is that they are linked to the browser relatively, which means that they are included in cache mechanisms of browsers. Obviously this creates more lightweight sites and pages, thus improving response times and reducing data-traffic.

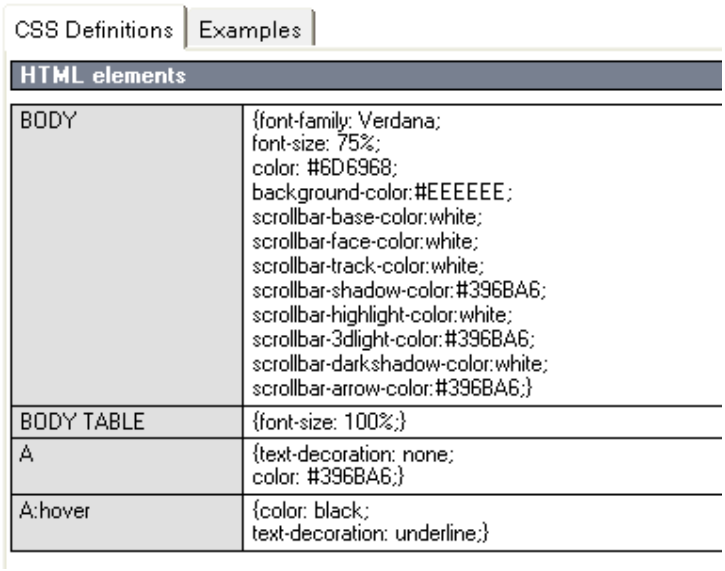
Key for this document	The key by which this JavaScript document is referenced.
Description	General description of the functions.
Content	The JavaScript routines being used.
Layout definition	The JavaScript routines being used obviously need to be included on the pages. The so-called binding of these pages is done in the Layout Definition (more relevant information can be found in that chapter).

Example usage for JavaScripts are:

- View formatting with custom sort routines
- Sitemap creation
- Menu control

8.2 CSS documents

The CSS documents, short for Cascading Style Sheet, are used to store CSS style sheets. The advantage of this is that they are linked to the browser relatively, which means that they are included in cache mechanisms.


CSS Definitions | Examples

HTML elements	
BODY	{font-family: Verdana; font-size: 75%; color: #6D6968; background-color:#EEEEEE; scrollbar-base-color:white; scrollbar-face-color:white; scrollbar-track-color:white; scrollbar-shadow-color:#396BA6; scrollbar-highlight-color:white; scrollbar-3dlight-color:#396BA6; scrollbar-darkshadow-color:white; scrollbar-arrow-color:#396BA6;}
BODY TABLE	{font-size: 100%;}
A	{text-decoration: none; color: #396BA6;}
A:hover	{color: black; text-decoration: underline;}

Many Examples are given on the Example page. Obviously these need tuning for your specific situation.

8.2.1 Centering a page

To center the entire page, you can use the following CSS definition:

```
#Content
{
	text-align: center;
}
#ContentArea
{
	width: 700px;
	margin-left: auto;
	margin-right: auto;
	text-align: left;
}
```

8.3 JavaScript views

In Passage CMS it's possible to build up views in the browser. The Domino server generates JavaScript commands, which refer to JavaScript commands that are stored in a JavaScript library (linked to the view via the layout definition).
For example each view entry triggers a vE function that appends the passed data to an array. In the 'after the body' HTML field of the layout a finish command in JavaScript is given, and the data is written to the browser.

Javscript views available

The following views are currently available:

1.Library JS
2.What´s new JS **Fallbacks:**
The source is no longer visible, previous and next buttons will no longer work.
Download:
JavaScript files that format views can be downloaded from our site www.passage.nl

Sample JS View:

```
var aViewEntries = new Array();

function TDB( className ) { document.write( "<TD class=\"" + className + "\">" ) }
function TDE() { document.write( "</TD>" ) }
function TRB( className) { document.write( "<TR class=\"" + className + "\">" ) }
function TRE() { document.write( "</TR>" ) }

function openDoc( docID ) {
// Opens the document given by DocumentUniqueID in read mode
document.location = DBNameHTML + "/0/" + docID + "?OpenDocument";
}

function viewEntry( docTitle, docID, docDepartment, docPhoneNumber, docFaxNumber,
docEmail, docPDate ) {
// Object constructor
// Refer to this function to see which data is available in the view
this.docTitle = docTitle;
this.docID = docID;
this.docDepartment = docDepartment;
this.docPhoneNumber = docPhoneNumber;
this.docKeywords = docFaxNumber;
this.docEmail = docEmail;
this.docPDate = docPDate;
}


function vE( docTitle, docID, docDepartment, docPhoneNumber, docFaxNumber,
docEmail, docPDate ) {
// Create an object instance in the array
oViewEntry = new viewEntry( docTitle, docID, docDepartment, docPhoneNumber,
docFaxNumber, docEmail, docPDate );

if (aViewEntries.push )
```

```
pushed = aViewEntries.push( oViewEntry )
else
aViewEntries = aViewEntries.concat(oViewEntry )
}

function writeTable() {
// Write out the final HTML

// Number of columns to be used
var iColumns = 1;

// Writing script
var i = 0;
var className1;
var record
var iAViewEntriesDivider = aViewEntries.length / iColumns;
if( iAViewEntriesDivider == 0 ) iAViewEntriesDivider = 1;
if(aViewEntries.length==0) i = 1;

for( var iColumnNumber = 1; iColumnNumber<=iColumns;iColumnNumber++) {
TDB( 'TData'+iColumnNumber);
// document.write(iColumnNumber+'<BR>');

for( i = 0; iColumnNumber * iAViewEntriesDivider > i; i++ ) {

record=i+1;

if (i & 1)
className1 = "Odd"
else
className1 = "Even"

document.write( '<DIV class="' + className1 + '"><FONT class="TitleSmall"><A
href="JavaScript:openDoc(\" + aViewEntries[ i ].docID +'\')" class="Hyperlink">' +
aViewEntries[ i ].docTitle + '</A></FONT><FONT class="Summary"><B>e-mail:</B> ' +
aViewEntries[ i ].docEmail + '</FONT><IMG src="/icons/ecblank.gif" width=5
height=5><FONT class="Summary"><B>Phone number:</B> ' + aViewEntries[ i
].docPhoneNumber + '</FONT></DIV><IMG src="/icons/ecblank.gif" width=5
height=5><HR>');

}
TDE();

}

}
```

Please note:
JavaScript is a programming Language. You do need Programming Skills to modify JavaScript code. When using JavaScript views you do need extensive knowledge on javaScript. Numerous resources are available on the Web.

The examples provided are as is and are no part of the Passage CMS product. Support on JavaScript can be given by Your Passage based on hourly rates.

9 Mail-In Filters

9.1 Introduction

Domino is a platform that allows you to send e-mails to a database. This means that you can also send e-mails to your Passage CMS database.

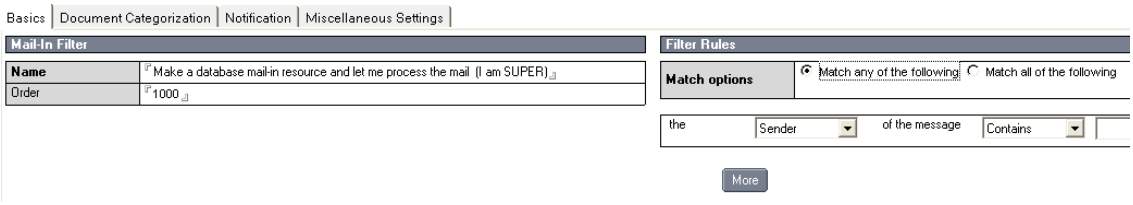
The handling of these e-mails is taken care of by an agent in Passage CMS that uses "Mail-in Filter documents" to categorize and publish the e-mails that have been sent. See it as "converting an e-mail to a Passage CMS document".

Each Mail-in Filter document has rules that you can define yourself – if the rules match, that Mail-in Filter document will be used to process the e-mail.

The e-mail will always be converted to a "Library" document. All fields that are present when creating a Library document by hand, can also be set on the Mail-in Filter document, so e-mails that are processed will be turned into Library documents with the values as set on the Mail-in Filter.

For each Mail-in Filter you can define:

- Filter name, order and rules
- Document categorization settings
- Notification settings
- Miscellaneous settings – add the date to the title, behaviour when a document with the same title already exists



There are two types of mail-in, filtered mail-in and structured mail-import.

9.2 Defining Mail-In Filters

Configuration	
Name	Fill in the name of the mail-in filter
Order	Order number, which one must be handled at first
Filter Rules	It can consist out of more than one filter rule. Fields that can identify the information source are: - Sender - Subject - To fields
Document categorization	The mail-in filter can automatically fill in several passage fields.
Audience	Here you can specify the audience by clicking the names of Persons and Groups existing in the Address Book.
Information category	Choose the Library Information Category you want this e-mail to be stored in
Keyword / Meta keywords	Choose a keyword as defined in the keyword libraries document at Library level
Icons	
Status	Choose a icon from the image store "Passage icons"
Validity	This will set the document into the chosen state. The validity chosen here will occur in the library document in the bottom field
Notification	A Mail-In filter contains the functionality to send notifications. It can send a mail to an e-mail address. Also it can send notifications to the Next editor via Passage.
E-mail notification	Fill in the "Notification address" and set "notification by e-mail" to **Yes**.
Next editor Passage	Fill in the "Next editor"
Miscellaneous Settings	The name of the form or view to which the definition applies.
Always prepend the current day to the title	Prepending the date to the title can make it easier to see what's new in a view

What should be done if a document has the same title and is in the same Library Information Category as an existing document?
The word 'slower' behind 'Keep existing document' and 'Delete existing document' means this task asks a bit more performance from the server compared to the 'Nothing' option.

Nothing' means: publish the document the way it is, so there will be two different documents with the same name in the same Library Information Category.

9.3 Structured mail import

Structured mail import is used in projects where the sender of the mail is under control of the Passage database owner. Using this functionality, the sender can be

programmed in great detail on how the mail should be formatted, and where the e-mail is stored in the Passage database.

Items to be programmed in the mail are:

- Library category
- Keywords
- Validity
- Workflow status
- Sender id
- Current and Next editor
- Target body field (e.g. rich-text or text)
- Title
- Overwrite function
- Special fields

Please contact Your Passage for additional information on this functionality.

10 Document life cycle management

Passage has an agent that periodically searches the database for expired and revised documents. This agent generates an e-mail message to all editors in chief of the database. Please ask your system administrator to enable this agent.

11 Customizing Passage code

Never alter the design of the Passage database without written confirmation of Your Passage. You are in violation of copyright laws if you modify Passage code.

After authorisation it is possible to customise Passage, for example to:

- Store extra keywords on a document, and do some calculations
- Create custom search screens on the web Modifying Passage design will mean that automatic upgrade becomes problematic. To prevent this, custom code can be put into subforms loaded on the documents.
- Create related documents or input/order/... forms to have a content centric experience

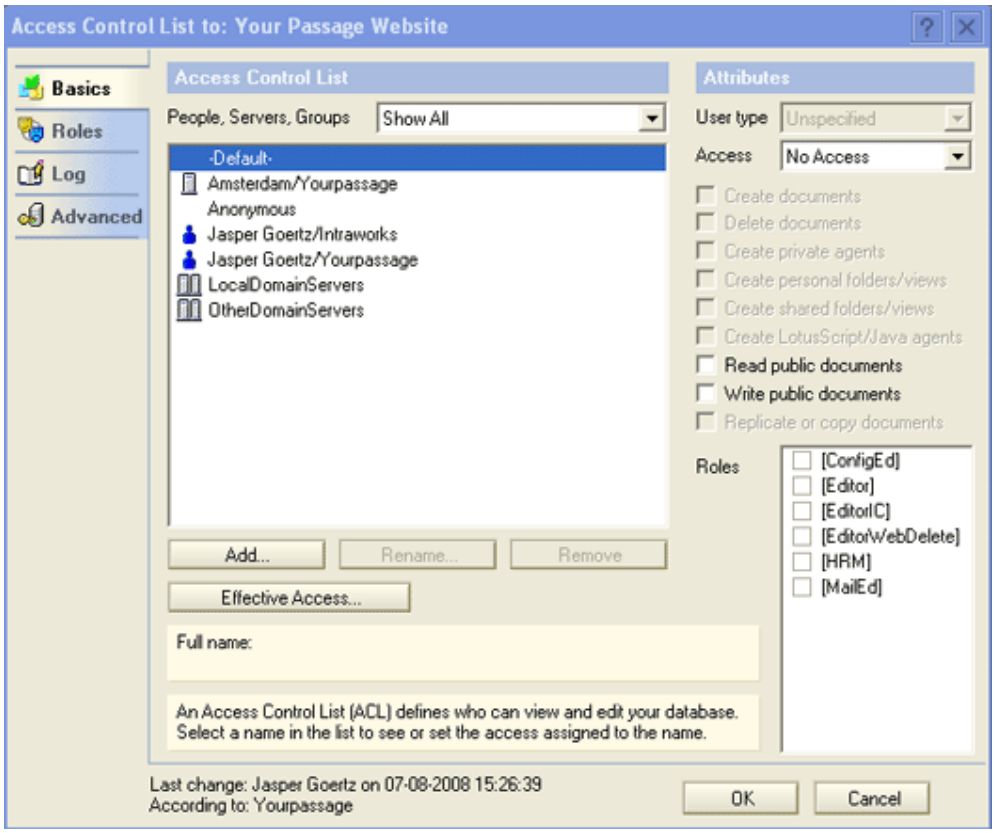
Custom Subforms that you are allowed to change are:

- Custom Fields - Address book | s5001BBA1
- Custom Fields - Brochures | s5006BBA1
- Custom Fields - Events | s5007BBA1
- Custom Fields - FAQs | s5004BBA1
- Custom Fields - Library Bottom | s5000BBA1
- Custom Fields - Library Top | s5000BBA2
- Custom Fields - News | s5002BBA1
- Custom Fields - Photographs | s5003BBA1
- Custom Fields - Presentations | s5009BBA1
- Custom Fields - Testimonials | s5010BBA1
- Custom Fields - Text Page | s5008BBA1
- Custom Fields - Text Page Top | s5008BBA2
- Custom Fields - Vacancies | s5011BBA1

12 Configuring the Database / Security

Passage workflows are configured using the ACL (access control list) of the database. The Administrator (database manager) assigns roles to employees in the ACL

In the ACL these roles should exactly be spelled as the below image. Notice the use of capitals! If the roles are not spelled correctly some of the Passage functions won't work. The default access level for the Passage database in Notes is **Author**.



The **MAXIMUM** access level for the Passage database for web access is **Author**.

The Configuration Editor (**ConfigEd**) has access to the Passage database configuration settings and the authority to change these. The access level for this role must be **Editor.**

ConfigEd	Configuration Editors can change the configuration of the Passage CMS database.
Editor	Editors (**Editor**) can create documents and concept versions and forward these to the Editor in Chief.
EditorIC	The Editor in Chief (**EditorIC**) can approve documents, publish them and "unpublish" them.
EditorWebDelete	Extra Editor role that can be used (activate on the administration document) to limit the delete function in the web edit environment.
MailEd	Mail Editors (**MailEd**) are authorised to read incoming mail and to forward it to other Editors.
HRM	People with the role of Human Resource Manager (**HRM**) are authorised to read job applications that come in and to process them. For additional server settings, please refer to the Passage Administrators Manual.